SVEUČILIŠTE U RIJECI
FILOZOFSKI FAKULTET U RIJECI
Odsjek za politehniku

Ružica Lončar

Interdisciplinarnost kod mobilnih robota

(završni rad)

Rijeka, 2018. godina

SVEUČILIŠTE U RIJECI
FILOZOFSKI FAKULTET U RIJECI

Studijski program: Sveučilišni preddiplomski studij politehnike

Ružica Lončar
JMBAG : 0009072108

Interdisciplinarnost kod mobilnih robota

-završni rad-

Mentor: mr.sc. Gordan Đurović, v.pred.

Rijeka, 2018. godine

FILOZOFSKI FAKULTET U RIJECI

Odsjek za Politehniku

U Rijeci, 10. studeni 2017. godine

ZADATAK ZA ZAVRŠNI RAD

Pristupnik: **Ružica Lončar**

Studij: **Sveučilišni preddiplomski studij politehnike**

Naslov završnog rada: **Interdisciplinarnost kod mobilnih robota / Interdisciplinarity in mobile robots**

Znanstveno područje: **2. Tehničke znanosti**
Znanstveno polje: **2.03. Elektrotehnika**
Znanstvena grana: **2.03.06 automatizacija i robotika**

Kratak opis zadatka: Na primjeru mobilnog robota IntelliBrain-Bot tvrtke RidgeSoft, opisati interdisciplinarnost mobilnog robota kao sustava. Opisati djelove mobilnog robota (motori, senzori, elektronika) i postupak njegovog sastavljanja, principe rada i sustav za programiranje uz isticanje intedisciplinarne povezanosti u svakom pojedinom segmentu. Opisati različite moguće pristupe navigacije mobilnog robota IntelliBrain (odometrija, korištenje orijentira, upotreba video signala) s aspekta interdiciplinarnosti u njihovoj primjeni. Na primjeru programa u JAVA programskom jeziku prikazati mogućnost učenja programiranja kroz stupnjevanje težine povezanih zadataka te interdiciplinarne veze s robotom kao jedinstvenim sustavom.

Zadatak uručen pristupniku: **10. studeni 2017. godine**

Ovjera prihvaćanja završnog rada od strane mentora: _____

Završni rad predan: _____

Datum obrane završnog rada: _____

Članovi ispitnog povjerenstva: 1. predsjednik -_____

2. mentor - _____

3. član - _____

Konačna ocjena: _____

Mentor

_____

mr. sc. Gordan Đurović

IZJAVA

Izjavljujem da sam završni rad izradila samostalno, isključivo znanjem stečenim na Odsjeku za politehniku Filozofskog fakulteta u Rijeci, služeći se navedenim izvorima podataka i uz stručno vodstvo mentora mr.sc. Gordana Đurovića.

U Rijeci , 14.09.2018.

SAŽETAK

Edukacijski mobilni robot naziva IntelliBrain-Bot ostvaruje interdisciplinarnost kao pojam koji povezuje razne discipline u jedan cjeloviti sustav. Razni elementi mobilnog robota ostvaruju interdisciplinarnost tako da povezuju mehaničke i elektroničke dijelove primjenom raznih senzora, priključaka, upravljačkog kontrolera, primjenom difrencijalnog sustava upravljanja kotača i dr. Također, interdisciplinarnost povezuje navedene elemente sa unaprijed određenim programskim sustavom RoboJDE koji koristi programski jezik Javu. Unutar navedenog programskog sustava se očituje uspješna povezanost svake klase, metode i funkcije te svakog elementa mobilnog robota. Primjer korištenja interdisciplinarnosti se očituje kod navigacije koji povezuju različite discipline kao npr. matematike, informatike te elektronike. Isto tako u primjeru povezanih zadataka gdje se zadatci lako modificiraju ostvaruje se interdisciplinarnost kao sustav unutar mobilnog robota.

KLJUČNE RIJEČI : mobilni robot, interdisciplinarnost , senzori , priključci, upravljački kontroler, diferencijalni sustav upravljanja, RoboJDE program, navigacija , modifikacija

U Rijeci, 14.09.2018.

SUMMARY

The educational mobile robot named IntelliBrain-Bot realizes interdisciplinarity as a concept that links various disciplines to one complete system. Different elements of mobile robots achieve interdisciplinary by linking mechanical and electronic components using various sensors, connectors, robotic controller, the use of a differential steering wheel system and etc. Also, interdisciplinarity binds these elements to a predefined RoboJDE programming system which using the Java programming language. Within the aforementioned program system, the successful connection of each class, method, and function of each element of the mobile robot is manifested. An example of using interdisciplinarity was manifested in navigation that links different disciplines such as mathematics, computer science and electronics. Likewise, in the example of related tasks where tasks can be easily modified, interdisciplinarity is realized as a system within a mobile robot.

KEY WORDS: mobile robot, interdisciplinarity, sensors, ports, robotic controller, differential steering wheel system, program RoboJDE, navigation, modification

In Rijeka, 14.09.2018.

SADRŽAJ

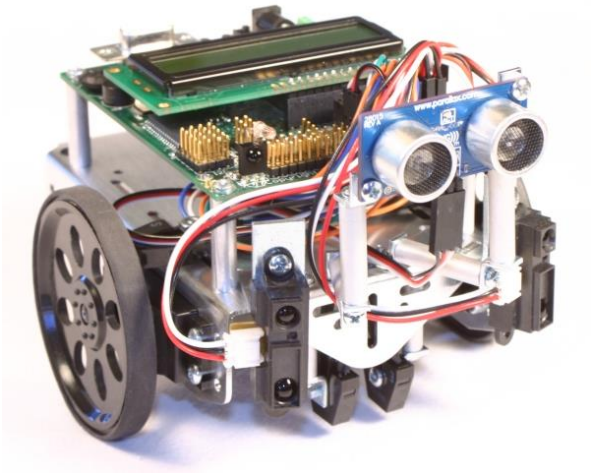
1. UVOD

Čovjek već tijekom povijesti želi izraditi mehaničku verziju samog sebe. Prvi put se dogodilo u 16.st. kada je talijanski izumitelj Leonardo da Vinci izradio mehanički automat odnosno mehaničkog lava. Također, pisci znanstvene fantastike su uspješno predvidjeli razvoj tehnike. Prvi put riječ robot se spominje u znanstvenom-fantastičnom djelu drami češkog pisca Karela Čapeka naslova „Rur“ odnosno „Rossumovi univerzalni roboti“. Kasnije je iz češke riječ „Robota“ izveden izraz robot što je značilo težak, prisilan rad. Isaac Asimov, pisac znanstvene fantastike, je riječ „Robotika“ uveo u svoje priče o uređajima, robotima što pomažu ljudima. Također, iznio je 3 zakona robotike 1942.godine koje govore da robot ne smije ozlijediti ljudsko biće niti zbog svoje neaktivnosti dopustiti da ljudsko biće pretrpi štetu, mora slušati naredbe osim kada te naredbe krše prvi zakon te robot mora štiti sebe osim ako bi to kršilo prvi i drugi zakon robotike. U današnjem vremenu robotika je veoma razvijena u razvijenim zemljama kao što su SAD, Japan, Kina itd. U tim zemljama sve više se koristi roboti da olakšaju ljudima kao što su rad u tvornicama odnosno gdje god se ljudski rad može zamijeniti sa robotom. Također, imamo edukacijske mobilne robote koji simuliraju rad većih i jačih robota. Od simulacija mogu imati navigaciju, okretati se, kretati se u bilo kojem smjeru itd. Tako dolazimo do definicije robota da su elektromehanički uređaji koje ljudi mogu programirati i takvi mogu izvršavati svoj program dok ih ljudska naredba ne zaustavi.

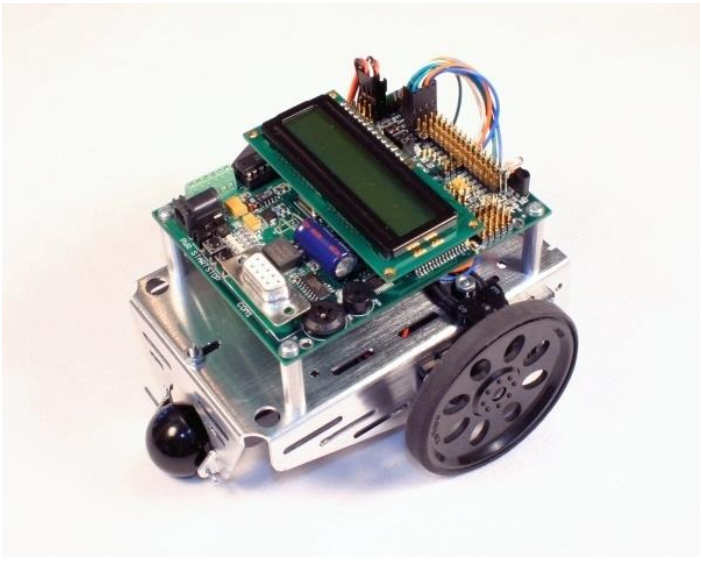
Cilj ovog završnog rada je opisati robot tvrtke RidgeSoft, njegove komponente i princip rada te opisati interdisciplinarnost mobilnog robota kao sustava uz isticanje interdisciplinarne povezanosti u svakom pojedinom segmentu. Također, opisati različite vrste metoda navigacija kao što su odometrija, navigacija pomoću orijentira i navigacija uz uporabu video signala povezujući pojam interdisciplinarnosti u njihovoj primjeni. Na kraju završnog rada opisati primjer programiranja u programskom jeziku Java povezajućih zadataka koje se na prethodni zadatak samo modificiraju te opisati interdisciplinarne veze s mobilnim robotom.

2.OSNOVNO O INTELLIBRAIN ROBOTU

IntelliBrain-Bot mobilni robot je edukacijski robot tvrtke RidgeSoft. Mobilni robot koristi programski jezik Java a programsku okolinu s kojim se kod prenosi u robot je program RoboJDE. Takav edukacijski mobilni robot omogućuje upoznati i razumjeti osnovne koncepte robotike, inženjerstva te informatičkih znanosti te tako ostvaruje pojam interdisciplinarnosti. Mobilni robot tvrtke RidgeSoft ima dva različita modela. Prvi takav model je IntelliBrain-Bot robot Deluxe opreme kojeg će biti opisan u sljedećim poglavljima (slika 2.1). Drugi model koji se razlikuje po vrsti robotičkog kontrolera i broju senzora je IntelliBrain-Bot Basic oprema (slika 2.2)



Slika 2.1 IntelliBrain-Bot robot deluxe oprema



Slika 2.2 IntelliBrain-Bot robot basic oprema

3. GRAĐA MOBILNOG ROBOTA

Edukacijski interdisciplinarni mobilni robot IntelliBrain-Bot dolazi u kit-kompletu koji omogućuje svakom kupcu sklapanje robota po uputama danim od strane tvrtke RidgeSoft. U takvom kit-kompletu dolaze pojedinačni dijelovi robota kao što su:

1) LCD ekran

2) IntelliBrain 2 upravljački kontroler

3) kučište za bateriju

4) dva servo motora

5) dva velika kotača

6) dva infracrvena fotoreflektivna senzora

7) kuglasti kotač

8) gumena brtva

9) četiri gumene vrpce za kotače

10) četiri metalna nosača kontrolera

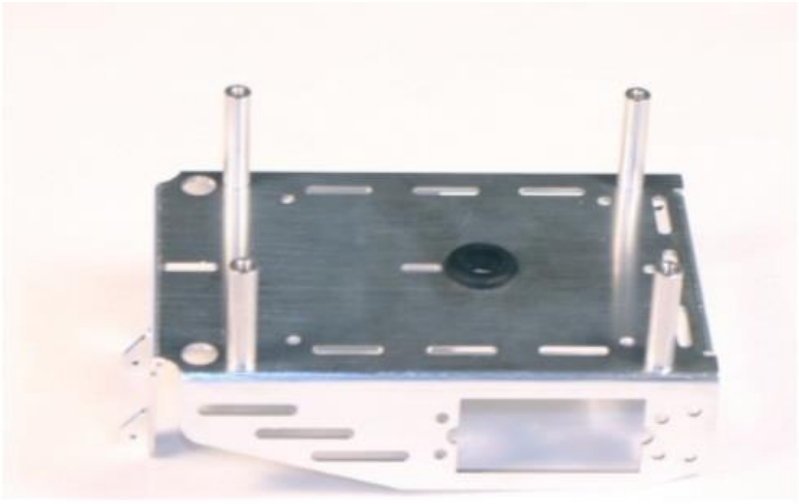
11) metalno kućište robota

12) metalna igla za kuglasti kotač

Uz navedene dijelove dolaze još osam kratkih vijaka s okruglom glavom, deset dugih vijaka s okruglom glavom, dvanaest matica te dva vijka s ravnom glavom (svi gore navedeni dijelovi se nalaze na slici 3.1). Također, u kit-kompletu se dobije četiri AA baterije od 1,5 [V] koje su potrebne za pokretanje robota, dodatni senzori te serijski kabel koji omogućuje spajanje robota s računalom. Da bi se robot mogao sastaviti potrebno je odgovarajući alat koji se nalazi u kit-kompletu a to je jedan obični i jedan križni odvijač i jedna špicasta kliješta.

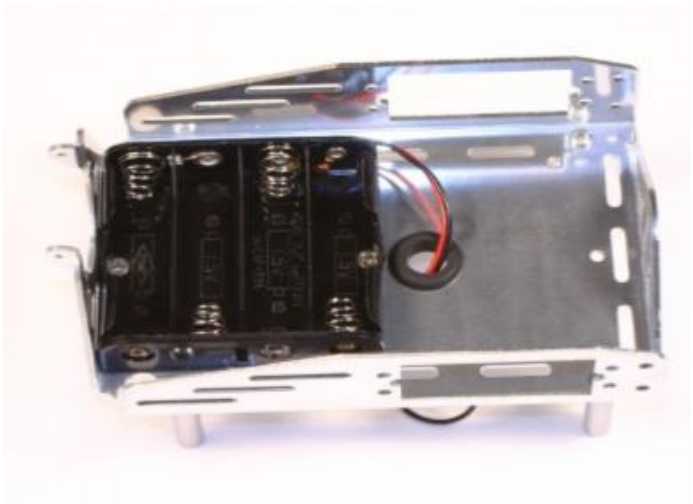
Slika 3.1 Dijelovi mobilnog robota u kit-kompletu

Proces sastavljanja IntelliBrain-Bot robota se sastoji od nekoliko postupaka. Za početak je potrebno iz kit kompleta pronaći metalno kućište i četiri kratka vijka sa okruglom glavom. Kada se navedeni vijci postave na metalno kućište dobiva se nosač za upravljački kontroler robota (prikazano na slici 3.2). Na prethodnoj slici je prikazana gumena brtva koja se ugura u otvor koji se nalazi na sredini kućišta robota. Namjena navedene brtve je zaštita izolacije žica da se ne ošteti tijekom struganja o rub kućišta. Na donju stranu kućišta se postavlja držač baterija. Za takvu montažu potrebni su dva dugačka vijka sa ravnom glavom koji se nalaze u kit kompletu. Navedeni vijci se postave s donje strane a potrebne se matice postave s gornje strane kućišta robota.

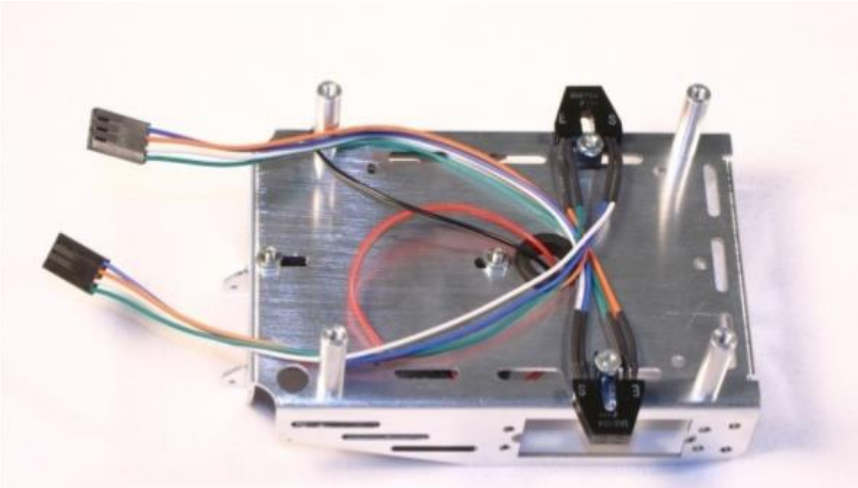


Slika 3.2 Prikaz nosača upravljačkog kontrolera

Potrebno je pripaziti pri montaži držača baterije na žice koje povezuju držač baterija s upravljačkim kontrolerom tako da ih se provuče kroz gumenu brtvu zaštićenog otvora u sredini kućišta robota (prikazano na slici 3.3). Na gornjoj strani nosača kontrolera robota postavljaju se infracrveni fotoreflektivni senzori tako da budu što više istureni preko ruba kućišta a koji služe za praćenje kretnje kotača robota (slika 3.4). Za postavljanje takvog senzora potrebna su dva dugačka vijka s okruglom glavom te dvije matice.



Slika 3.3 Prikaz montaže držača baterije postavljene na donjoj strani kućišta robota



Slika 3.4 Prikaz montaže infracrvenog fotoreflektivnog senzora

Da bi se robot kretao po ravnoj podlozi potrebno mu je dva kotača i dva servo motora. Na donjoj strani kućišta robota postavljaju se dva servo motora koji se svaki učvršćuje sa četiri vijaka s okruglom glavom i četiri matice koje će se nakon navedene montaže nalaziti s unutarnje strane kućišta robota. Nakon što se postave servo motori potrebno je pripremiti kotače. Tako da se na svaki kotač navuče po jedna gumena vrpca. Poslije pripreme kotača potrebno je postaviti kotač na nosač na servo motor s pomoću dva crna vijka. Velikim kotačima potreban je jedan mali kuglasti kotač koji se postavlja s stražnje strane robota između nosača na kućištu koji se učvršćuje pomoću metalne igle (prikazano na slici 3.5).

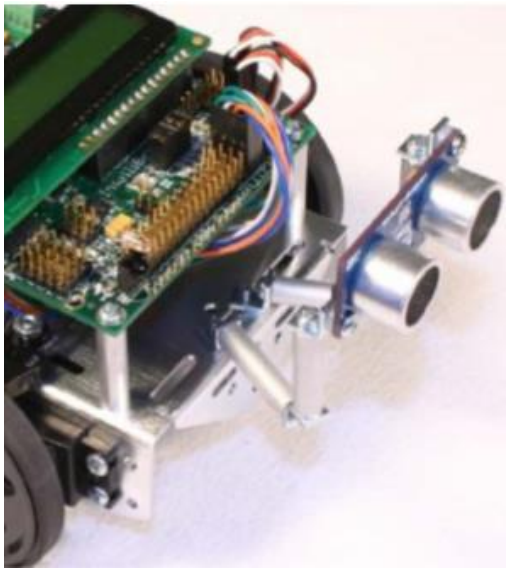
Slika 3.5 Prikaz nakon montaže servo motora, kotača i kuglastog kotača

Na prednju stranu robota postavljaju se senzori za mjerenje udaljenosti u kojoj se robot nalazi u odnosu na neki predmet ispred njega. Prije postavljanja na kućište robota potrebno je na nosaču u obliku slova L učvrstiti senzore pomoću malih vijaka s okruglom glavom, maticama i podloškom. Takve pripremljene senzore se učvršćuje na prednju stranu kućišta robota pomoću malih vijaka s okruglom glavom te maticama (slika 3.6. ispod a). U bijele konektore na boku senzora se dodaju odgovarajući konektori sa žicama koje služe za povezivanje senzora sa upravljačkim kontrolerom robota (slika 3.6 ispod b).

Slika 3.6 a) Postavljanje L nosača na senzore udaljenosti

Slika 3.6 b) Prikaz robota nakon montaže senzora udaljenosti

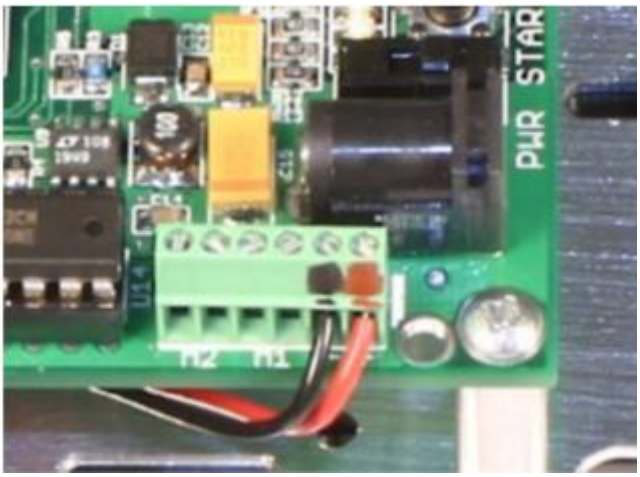
Na kućište robota se postavlja pločica sa ultrazvučnim senzorom s nosačima pomoću dva mala vijka na način da se vijci umetnu kroz rupe s donje strane kućišta robota a matice dolaze s gornje strane (slika 3.7).



Slika 3.7 Prikaz robota nakon montaže ultrazvučnih senzora

Glavna komponenta u procesu sastavljanja robota je upravljački kontroler koji se postavlja na kućište robota na prije postavljenom nosaču. Tijekom stavljanja kontrolera na nosače potrebno je okrenuti kontroler tako da je serijski port za spajanje na računalo bude na

stražnjoj strani robota. Da bi se pričvrstio upravljački kontroler za nosač potrebna su četiri kratka vijka s okruglim glavama. Kada se upravljačka ploča postavila na nosače potrebno je žice koje izlaze iz držača baterije povezati s upravljačkim kontrolerom na svjetlo zeleni priključak na lijevoj strani kontrolera pazeći na polaritet tako da ide crvena žica u utor označeno s crveno i oznakom „+“ te crna žica u utor označeno s crno i oznakom „- “. (slika 3.8).

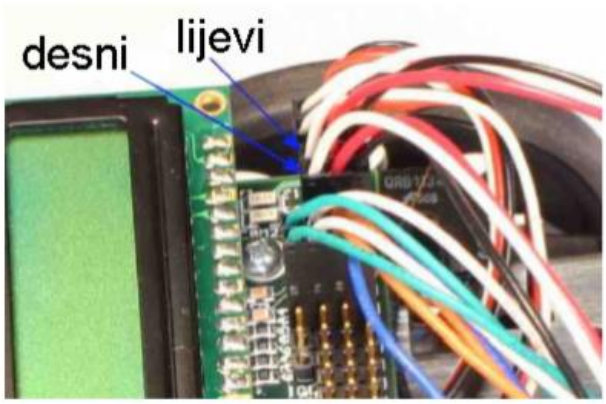


Slika 3.8 Prikaz povezivanja žica držača baterija na upravljački kontroler

Na upravljački kontroler se spajaju i servo motori tako da žice koje izlaze iz servo motora uz pomoć gumene brtve dolaze do kontrolera. Žice koje dolaze s lijevog servo motora na kontroler se spajaju na servo port1 označen sa „S1“ na kontroleru tako da se crna žica spaja na pin koji se nalazi na rubu samog kontrolera (slika 3.9). S desnog servo motora žice se spajaju na upravljački kontroler na servo port2 s oznakom „S2“.

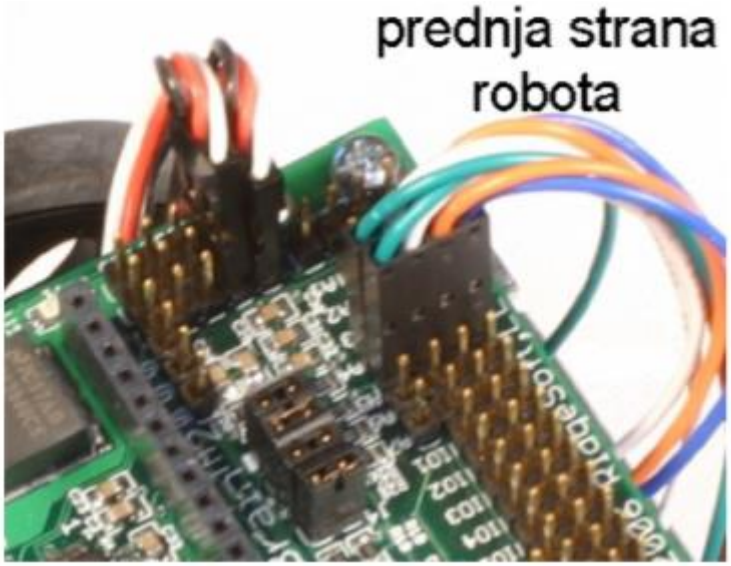
Slika 3.9 Povezivanje servo motora na upravljački kontroler

Također, na upravljački kontroler se spajaju senzori udaljenosti koji se povezuju na prednjoj lijevoj strani (slika 3.10). Na način, da se lijevi senzor spaja na port s oznakom „A1“ a desni senzor na port s oznakom „A2“.



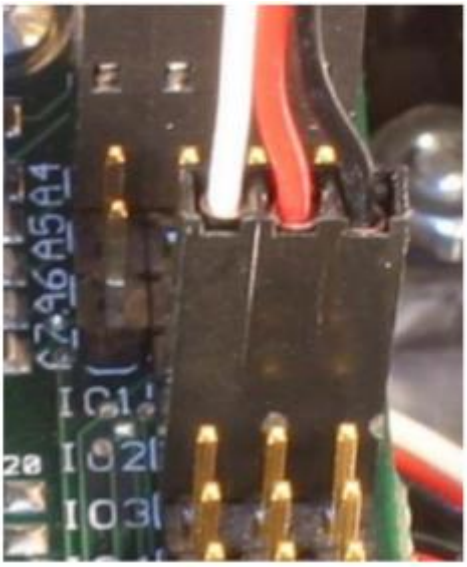
Slika 3.10 Spajanje senzora udaljenosti na upravljački kontroler robota

Infracrveni fotoreflektivni senzori se spajaju na prednju stranu kontrolera tako da žicu s lijevog senzora se spoji na analogni port 4 oznake „A4“ a žicu s desnog senzora na analogni port a5 oznake „A5“ i na način da se plave žice spajaju na pinove s ruba samog kontrolera (slika 3.11).

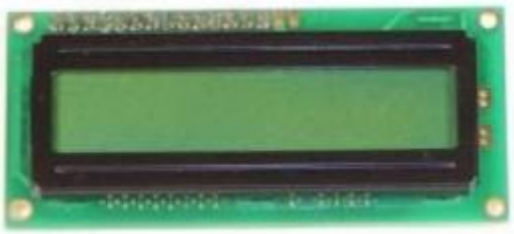


Slika 3.11 Spajanje senzora za kretanje kotača na upravljački kontroler

Posljednji senzor koji se povezuje sa upravljačkim kontrolerom je ultrazvučni senzor na način da žica se spaja na port označen „IO3“ tako da se crna žica postavlja na pin koji se nalazi najbliže prednjem rubu kontrolera (slika 3.12). Na kraju procesa sastavljanja robota se postavlja LCD ekran koji omogućuje dobivanje dodatnih informacija tijekom rada robota. Na donjoj strani LCD ekrana se nalaze 14 pinski konektor koji je potrebno spojiti sa 14 pinski konektorom na upravljačkom kontroleru robota (slika 3.13).



Slika 3.12 Spoj konektora ultrazvučnog senzora na upravljački kontroler robota

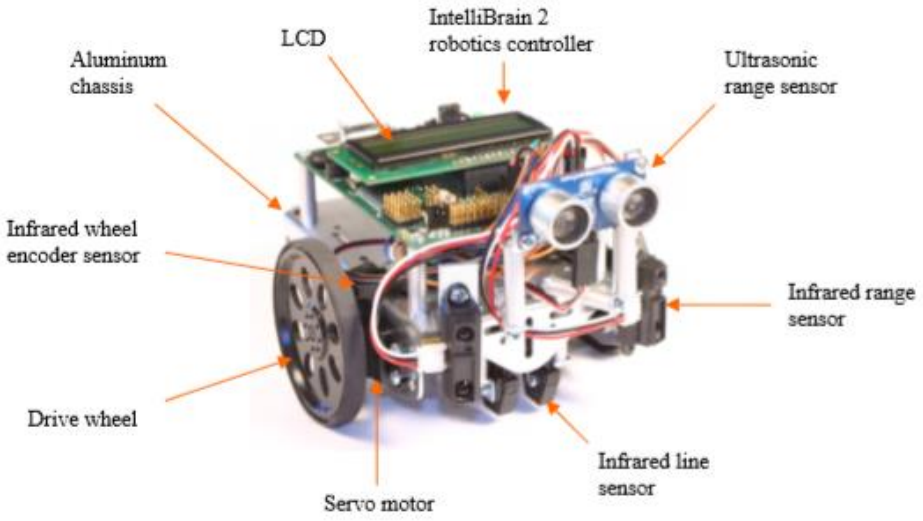


Slika 3.13 Izgled LCD ekrana

Nakon procesa sastavljanja komponenti robota dobili smo jedan sastavljeni IntelliBrain-Bot robot sa svim komponentima potrebnim za rad (pogledati poglavlje 2 , slika 2.1). Takvom mobilnom robotu se tijekom tehničkog sastavljanja mehaničkih i elektroničkih dijelova očituje pojam interdisciplinarnost kao povezivanje navedenih dvaju disciplina radi zajedničkog cilja a to je izrada funkcionalnog mobilnog robota IntelliBrain-Bot.

3.1. MEHANIKA ROBOTA

IntelliBrain-Bot robot upotrebljava jednostavni mehanički dizajn koji se sastoji od aluminijskog kućišta proizveden od jednog velikog komada metala. Takvo kućište omogućuje centriranu i čvrstu strukturu ovakvom robotu. Na donjoj strani metalnog kućišta nalazu se dva servo motora koja omogućuju da se dva kotača okreću svojom pogonskom snagom koju svaki kotač dobiva od jednog servo motora što znači da jednim kotačem upravlja jedan servo motor te također se na toj strani postavlja kućište za četiri AA baterije. Na drugoj strani kućišta imamo kuglasti kotačić koji drži zadnji dio kućišta mobilnog robota. Također, na gornjoj strani metalnog kućišta se montiraju pripadajući senzori i upravljački kontroler robota. Kod mehanike robota pojam interdisciplinarnosti se očituje u povezivanju različitih disciplina kao što su proizvodnja aluminijskog kućišta, elektronike upravljačkog kontrolera i mehaničkih dijelova kao što su servo motori, kotači, senzori i dr.

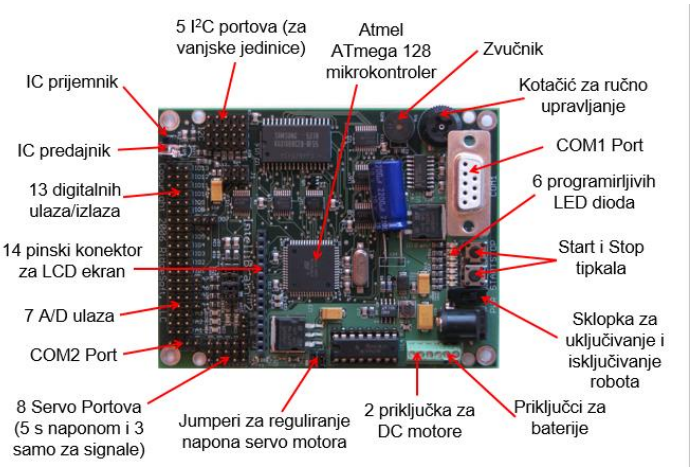


Slika 3.14 Detaljnija građa mobilnog robota tvrtke RidgeSoft

3.2. UPRAVLJAČKI KONTROLER

„Mozak“ mobilnog robota je upravljački kontroler (slika 3.15.) koji omogućuje da robot sam analizira senzorne ulaze, samostalno se kreće po omogućenom prostoru te da uz kontrolu motora ostvaruje napisani kod u programskom jeziku Java koji se nalazi unutar samog kontrolera. Program koji se kreira i učitava na upravljački kontroler može omogućiti da robot izvršava svakojake funkcije. Takvi programi mogu biti teški ili lagani ovisno o funkciji koju robot mora izvršiti. Također, programi mogu biti u rasponu od laganih do vrlo teških odnosno od napisanog imena na LCD ekranu pa do traženja i spašavanja robota u nevolji. Funkcije koje se mogu spremiti na upravljački kontroler mogu biti od samog kretanja naprijed robota ili okretanja robota 180 stupnja ili napisati poruku na LED ekranu robota itd.

Na upravljačkom kontroleru imamo mogućnost spajanja 23 senzora i krajnjih jedinica i uključujući dva servo motora. Središnja komponenta kontrolera je Atmel ATmega128 mikrokontroler frekvencije 14.7 MHz. Dodatak mikrokontroleru je vanjski RAM memorija koja ima 128 K bajtova na upravljačkom kontroleru. Također, takav mikrokontroler ima 128K bajtova na čipu FLASH memoriji koji ima može imati najviše 10,000 preuzimanja po danim specifikacijama.



Slika 3.15 Prikaz upravljačkog kontrolera

Na upravljačkom kontroleru interdisciplinarnog mobilnog robota imamo komponente koje omogućuju korisniku robota da lakše programira robota za interakciju s ljudima. Takvi komponenti su :

a) LCD ili ekran temeljen pomoću tehnologije tekućih kristala
   - ekran koji omogućuje od dva do 16 linija znakova
   - odličan je za pokazivanja rezultata tijekom korištenja senzora te omogućuje lakši način učenja o senzorima te učinkovitu dijagnostiku problema oko senzora
   - uključuje osnovno sučelje
   - način na koji se postavlja na upravljački kontroler pogledati u poglavlju 3. „Građa mobilnog robota“.
b) START i STOP dugme – omogućuje čovjeku da naznači mogućnosti na LCD ekranu ili da preuzet program započne ili se zaustavi ovisno o želji korisnika
c) THUMBWHEEL ili crno okruglo dugme – omogućuje pregled unutar sučelja na LCD ekranu te može mijenjati brzinu servo motora
d) BUZZER ili zvučni signal – komponenta koja daje zvučni signal a može biti trubljenje, šklocanje ili sviranje tona
e) LED diode – omogućuje vizualne pokazatelje korisnicima robota,
   - ukupno ih je sedam led dioda od kojih šest može biti uključeno, ugašeno ili žmigano a mogu biti tri crvene i tri zelene boje signala led dioda
   - prva lijeva led dioda pokazuje indikator napajanja te može dati zelenu boju signala led diode ako je uključeno

Sve navedene komponente su prikazane na slici 3.15. Kod upravljačkog kontrolera mobilnog robota IntelliBrain-Bot interdisciplinarnost se ostvaruje povezivanjem mehaničkih elemenata kao što su senzori i motori te programiranja kao jedinstvena disciplina koja se ostvaruje korištenjem navedenog programskog jezika Java. Također, interdisciplinarnost se očituje u interakciji mobilnog robota i čovjeka koji može koristiti različite gore navedene dugme na upravljačkom kontroleru.

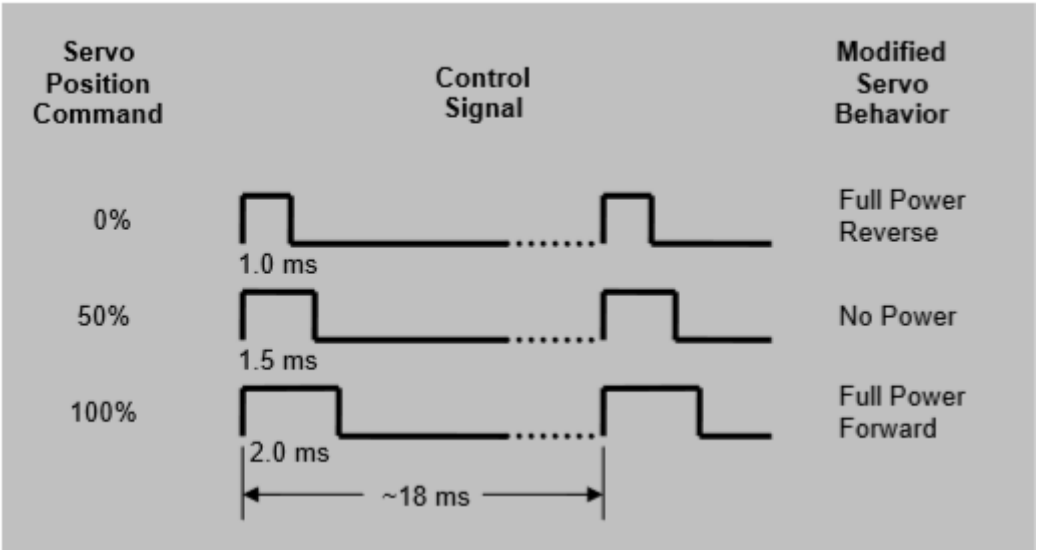
3.3. MOTORI

U mobilnom robotu tvrtke RidgeSoft kao što je navedeno u prijašnjim poglavljima i podglavljima se nalaze servo motori (slika 3.16) . Takvi motori imaju mehaniku dizajniranu za rotaciju servo izlazne osovine u određeni položaj i držanje trenutačnog položaja te ugrađenu kontrolu pokreta.

Slika 3.16 Servo motori

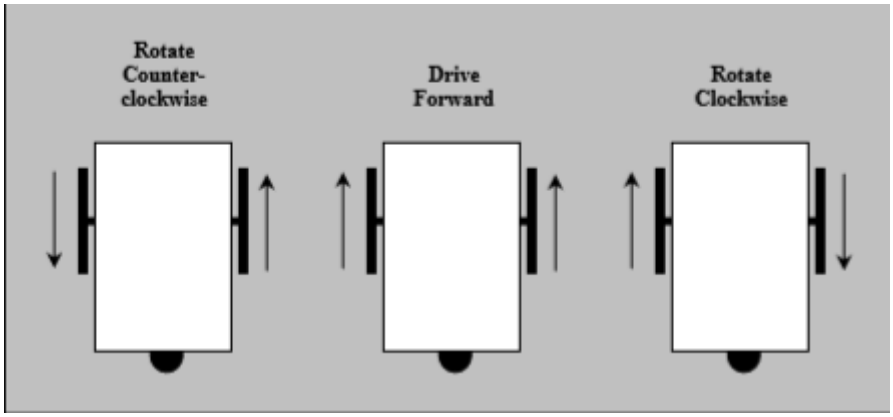
U kit-kompletu navedenog u poglavlju „ 3. Građa mobilnog robota“ se nalaze servo motori kontinuirane rotacije oni daju pogon kotačima robota. Takvi motori su pod nadzorom periodičnih pulsirajućih servo signala (slika 3.17). Obično su servo signali žuto ili bijele boje prikopčani na servo motor. Takve signale možemo objasniti uz pomoć paljenja i gašenja prekidača za svjetlo. Signali se mjere u milisekundama tako da u jednoj milisekundi signal diktira servu da se pomakne na jedan kraj raspona 0 % a to je jedan ekstrem.

Također, u drugom ekstremu se radi o dvije milisekunde i rotaciju na drugi kraj te raspona 100% . Da bi se robot mogao pomaknuti na određenu poziciju moramo biti unutar navedena ekstrema. Što znači, ako naredimo motoru poziciju 0% i dodamo milisekundu onda rezultat će biti kretanja kotača unatrag i maksimalan je okretni moment a ako postavimo položaj na 100% onda dobijemo maksimalni okretni moment te kretnju prema naprijed. A ako naredimo servo motoru položaj od 50% onda je okretni moment jednak nuli. Detalji oko pulsirajućih servo signala su već napisani od strane programa RoboJDE. Osim servo motora, mobilni roboti mogu biti pogonjeni DC motorima koji na sličan način funkcioniraju uz pomoć alata nazvanog engl. „IntelliBrain Expansion Board“.



Slika 3.17 Prikaz pulsirajućih servo signala

Kako servo motori pomiču kotače robota dolazimo do pojma sustava diferencijalnog upravljanja robota (slika 3.18) što znači da je svaki servo motor može imati svoju brzinu i svoj smjer. Mobilni robot dizajniran ovim sustavom ima dva neovisna pogonska kotača kako bi moglo upravljati njihovim smjerom i brzinom. Promjenom smjera i brzine robot može izvesti osnovne vještine kao što su vožnja unaprijed, okretanje u smjeru kazaljke na satu ili obratno. Da bi se robot kretao naprijed potrebno je postaviti na istu vrijednost snage na oba dva motora a ako bi se kotači robota mogli rotirati na mjestu potrebno je snagu na servo motorima postaviti na istu vrijednost ali obrnutog smjera. Okretanje u smjeru kazaljke na satu ili obrnuto će uspjeti kada se primjeni pravilo takvo da se postavi više snage na jedan motor a manje na drugi. Zbog toga će jedan kotač brže ići od drugog pa će se robot okretati u zavisnosti koji servo motor ima veću snagu. Mobilni robot IntelliBrain-Bot osim osnovnih vještina može raditi i sofisticirane vještine kao što su ponavljanja kretanja, kretanje u obliku kvadrata i svih drugih oblika geometrije. Sustav diferencijalnog upravljanja robota ovisi o bateriji mobilnog robota. Na način da trošenjem baterije dolazi do promijenjenog ponašanja mobilnog robota.



Slika 3.18 Prikaz diferencijalnog upravljanja kotača mobilnog robota

Servo motori obično se koriste za aplikacije namijenjene robotima zbog jeftine cijene pri kupnji, lagane konstrukcije, lagana ugradnja te jednostavno korištenje. Osim kod robota servo motori imaju dobru primjenu i kod letjelica (slika 3.19)

Slika 3.19 Prikaz letjelica koji koriste servo motore

Pojam interdisciplinarnosti kod servo motora mobilnog robota se ostvaruje povezivanjem periodičnih pulsirajućih servo signala koji daju pogon kotačima, diferencijalni sustav upravljanja te kotači koji pripadaju kao mehanički elementi mobilnog robota.

3.4. SENZORI

Informacije koje mobilni robot prikuplja o okruženju oko njega uspijevaju uz pomoć senzora koji se nalaze na mobilnom robotu. Svaki senzor ima koji stupanj izvjesnosti zbog mogućih više mjerenih elemenata. Mobilni motor ima dvije stvari za prikupiti iz senzora a to su:

a) svoje stanje – brzinu motora, napon baterije i dr.

b) njegovo cijelo okruženje

Općenito, senzori se dijele na dvije vrste :

a) aktivni senzori – za svoj postupak ne zahtijevaju dovođenje dodatne energije te na osnovu energije nastaje izlazni signal, mogu biti induktivni, elektromagnetski, elektrodinamički i dr.

b) pasivni senzori – potrebna ima je dodatna energija te za nekog vanjskog izvora modeliraju energiju, mogu biti induktivni, otpornički i kapacitivni

Tipovi pogrešaka koje se mogu pojavljivati tijekom primjene senzora za mobilne robota zbog razlika između mjerenih i stvarnih vrijednosti :

a) sistemske greške – pogreške koje se javljaju tijekom loše komunikacije senzora i robota ili se koristi samo jedan senzor za više istovremenih mjerenja vrijednosti

b) nasumične greške – pogreške prilikom neodređenog ponašanja senzora odnosno pogreške zbog sjajne površine, slabog svjetla te ponekih kalibracija

Na mobilnom robotu tvrtke RidgeSoft imamo sedam senzora a oni su :

a) ultrazvučni senzor

b) dva infracrvena fotoreflektivna senzora

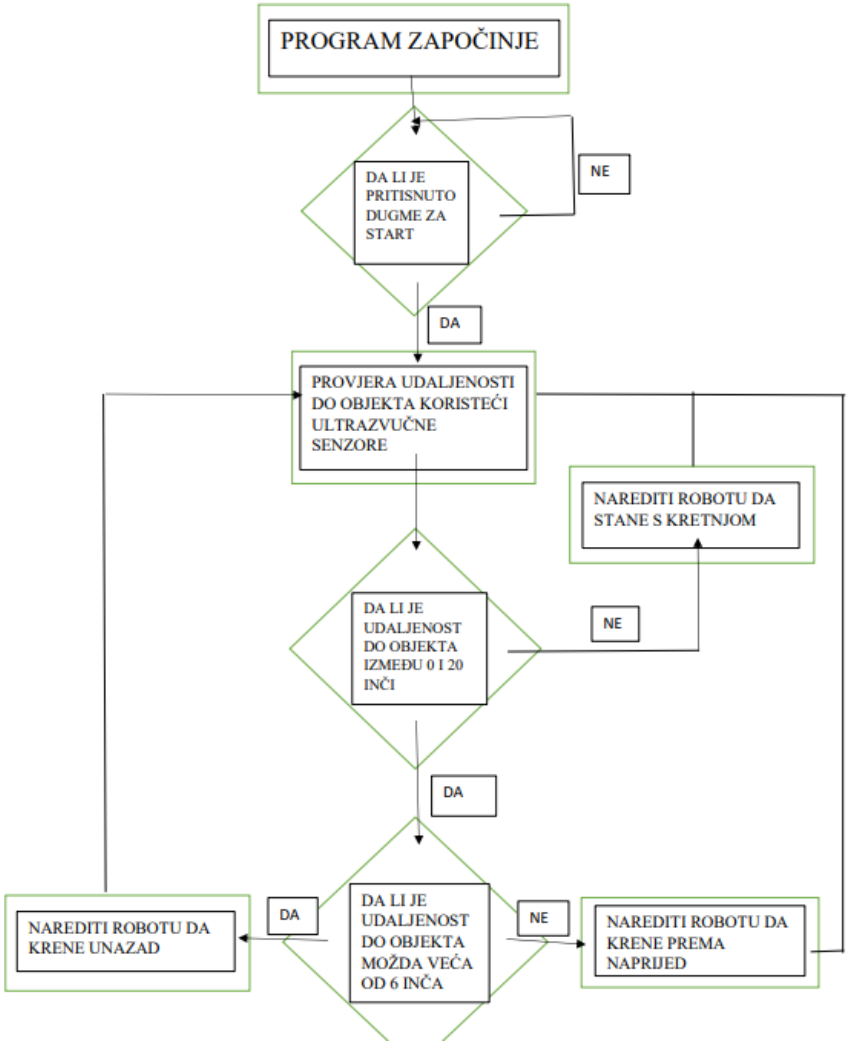
c) dva senzora sa praćenje kretanja kotača mobilnog robota

d) dva senzora za mjerenje udaljenosti

3.4.1 Ultrazvučni senzor

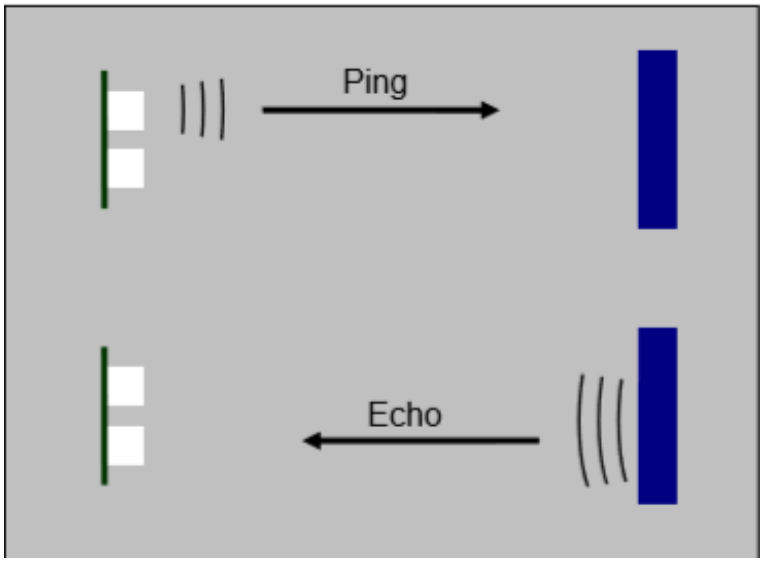
Senzor mobilnog robota (slika 3.20) mjeri udaljenost uz pomoć izdavanja kratkog impulsa visoke frekvencije do predmeta odnosno brzine zvuka i vremena koji je prošao do tog određenog predmeta odnosno radi na principu odbijanja vala zvuka od određenog objekta kojeg odašilje ultrazvučni transmiter a sve vrijednost ultrazvučni senzor preuzima. Što se tiče udaljenost omogućuju detekciju udaljenosti predmeta između 1.2 i 118 inča koji se nalaze ispred robota. Udaljenost do objekta se izračunava pomoću brzine vala zvuka i prijeđenog zvuka. Mobilni robot IntelliBrain-Bot će održavati udaljenost prema promatranom objektu koja neće biti veća od 20 inča a manja ili jednaka od 5 inča jer inače će mobilni robot će se udaljiti od promatranog objekta (slika 3.21).

Slika 3.20 Ultrazvučni senzor kod mobilnog robota



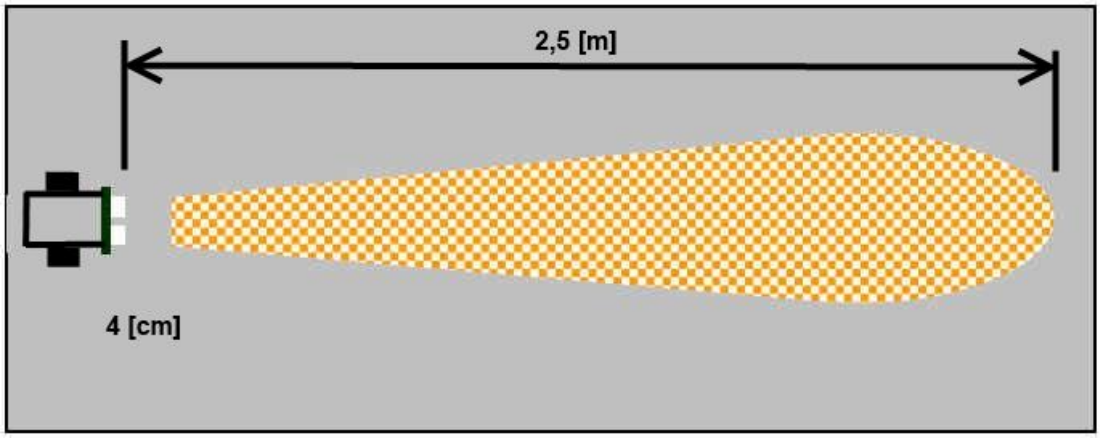
3.21 Primjer korištenja ultrazvučnog senzora

Ultrazvučni senzor koji koristi IntelliBrain-Bot robot se naziva Parallax Ping)))™ koji je prikazan na slici 3.20. Objašnjenje djelovanja ultrazvučnog senzora se može olakšati od navedenog teksta gore. Na način, da navedeni ultrazvučni senzor omogućuje mjerenje udaljenosti od objekta do prednje strane mobilnog robota gdje se ultrazvučni senzor nalazi uz pomoć engl. naziva ping. Senzor šaljući kratak impuls visoke frekvencije pronalazi objekt pa od objekta prima povratne impulse a ako ne pronađe objekt onda se neće pojaviti povratni impuls. Na takav način se može odrediti i udaljenost mjereći vrijeme između pojave impulsa i njegovog povratka do senzora (slika 3.22)



Slika 3.22 Mjerenje udaljenosti uz pomoć ultrazvučnog senzora

Udaljenost između ultrazvučnog senzora i objekta je u obliku konusa počevši od mobilnog robota. (slika 3.23).Dimenzije konusa su promjenjive ovisne o svojstvima površine podloge te svojstva pronađenog objekta.

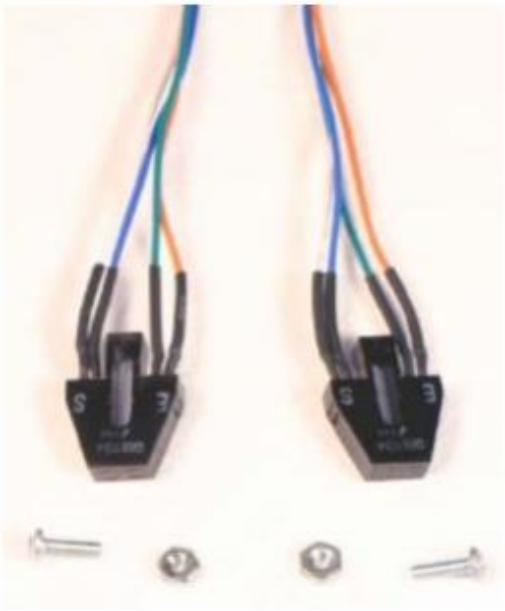


Slika 3.23 Prikaz udaljenosti u obliku konusa za ultrazvučni senzor

Ultrazvučni senzor je povezan s upravljačkim kontrolerom uz pomoć jednog od četiri ulazno/izlazno digitalnih priključaka naziva IO3, IO4, IO5, IO6. Detalje u vezi spajanja senzora na upravljačku ploču pogledati podglavlju „3.5 elektronika“ a detalje vezane za sastavljanje ultrazvučnog senzora pročitati u uvodu poglavlja „3. Građa mobilnog robota“.

3.4.2. Infracrveni fotoreflektivni senzor

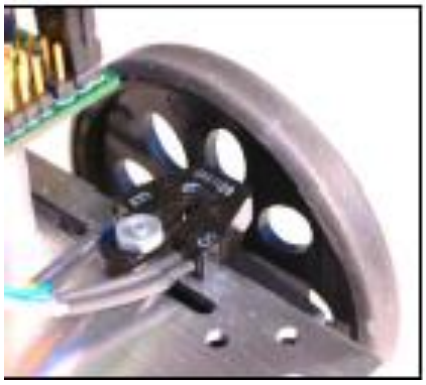
Ovaj senzor za mobilnog robota naziva QRB1134 služi za primjećivanje predmeta između 4 i 30 inča ispred robota. Također, mjere kutni odraz uske zrake infracrvenog svjetla da bi mogli izmjeriti raspon do predmeta. Takvi se senzori koriste za pronalaženje i izbjegavanje predmeta koji se nalaze na određenom putu koji robot mora proći.



Slika 3.24 Infracrveni fotoreflektivni senzori i njihovi elementi za spajanje na kućište robota

3.4.3. Senzori za praćenje kretanja kotača mobilnog robota

Omogućuju gibanje robotskih kotača i koriste fotoreflektivnu infracrvenu svjetlost kako bi mogli osjetiti rupice i krak na kotačima (slika 3.25). Takav signal pouspješuje provedbu shaft encodera koji u svakom trenutku poziciju kotača mobilnog robota. Također, robot poznavajući svaku poziciju kotača dolazi do pojma lokalizacije koja se spominje unutar poglavlja „5. Osnovno o navigaciji mobilnog robota“. Shaft encoder omogućuje mobilnom robotu vožnju sa jednog mjesta na drugi. Takva tehnologija se oslanja na očitavanje signala koji kodira kutnu poziciju vratila odnosno u slučaju ovog robota je osovina na koju je kotač ugrađen. Shaft encoderi se koriste kod odometrije, mjerača za brzinu , bicikli , motociklama te kod tahometara u automobilima (mjerača za brzinu automobila).

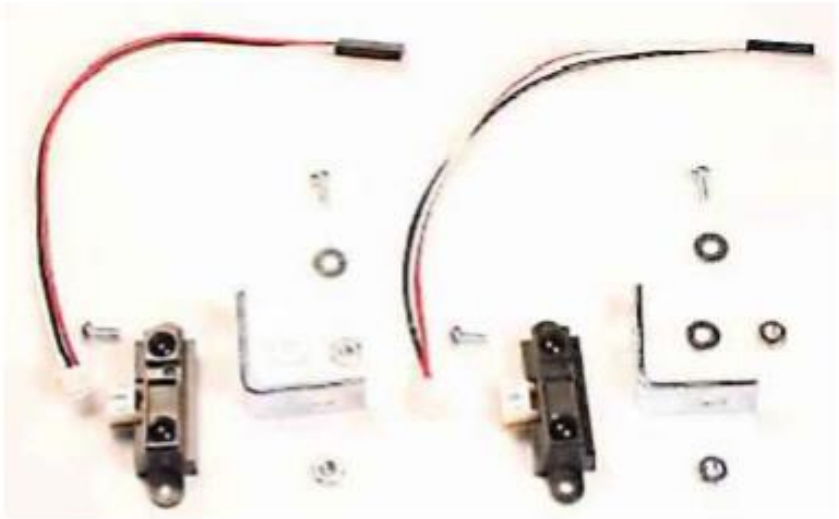


Slika 3.25 Prikaz korištenja infracrvenog fotoreflektivnog senzora za praćenje kretnje kotača mobilnog robota

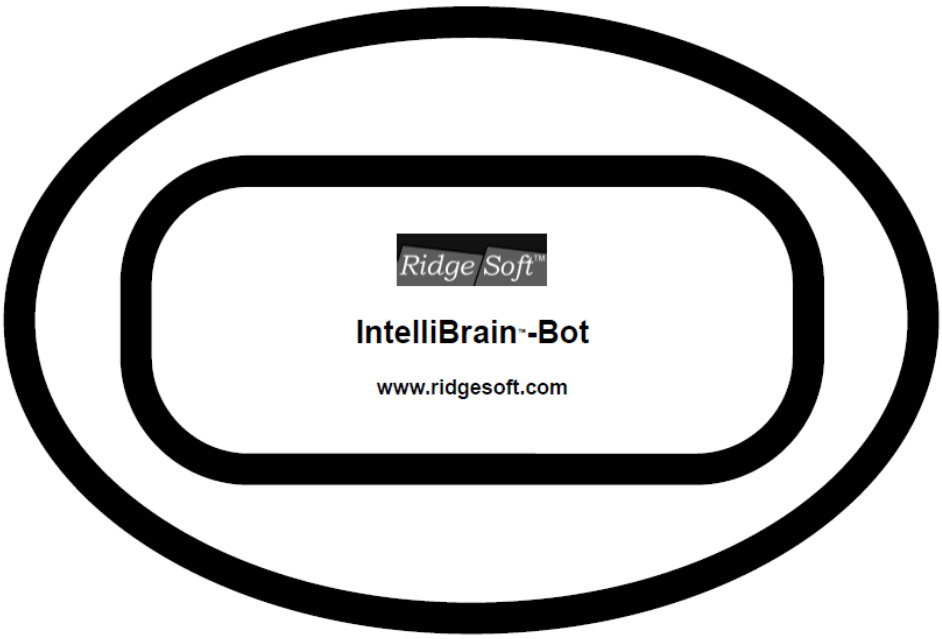
Mobilni robot IntelliBrain-Bot dolazi sa dva plastična kotača kao što se vidi na slici 3.25 . Svaki kotač se sastoji od osam krakova odjeljenih sa osam duguljastih rupa. Krakovi i rupe na kotaču su temeljni elementi za očitavanje kretnji na kotaču.

3.4.4. Senzori za mjerenje udaljenosti

Smještaj senzora za mjerenje udaljenosti se nalazi na donjoj prednjoj strani robota (slika 3.26). Takvi senzori koriste infracrvenu svjetlost da bi osjetili reflektirajuću podlogu ispod robota te omogućuju praćenje pravca ne reflektirajuće crne linije od jednog inča na visokoj reflektirajućoj bijeloj podlozi. Primjer takve podloge vidjeti sliku 3.27..



Slika 3.26 Senzori udaljenosti i dijelovi za njihovu montažu

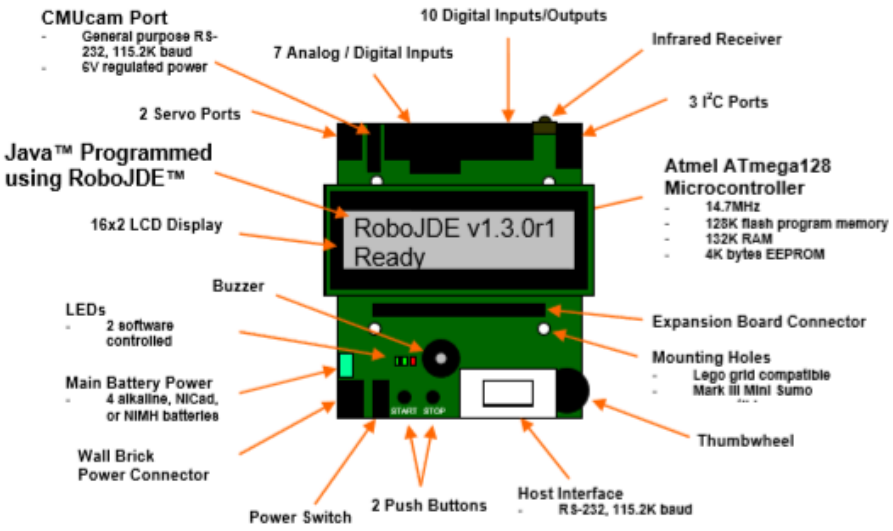


Slika 3.27 Prikaz podloge za uporabu senzora za praćenje pravca kretanja

Pojam interdisciplinarnosti kod senzora robota IntelliBrain-Bot se očituje u povezivanju izdavanog kratkog impulsa visoke frekvencije odnosno princip odbijanja vala zvuka, uske zrake infracrvenog svjetla sa upravljačkim kontrolerom mobilnog robota. Također, ostvaruje se i u povezivanju gibanja robotskih kotača i fotoreflektivne infracrvene svjetlosti senzora.

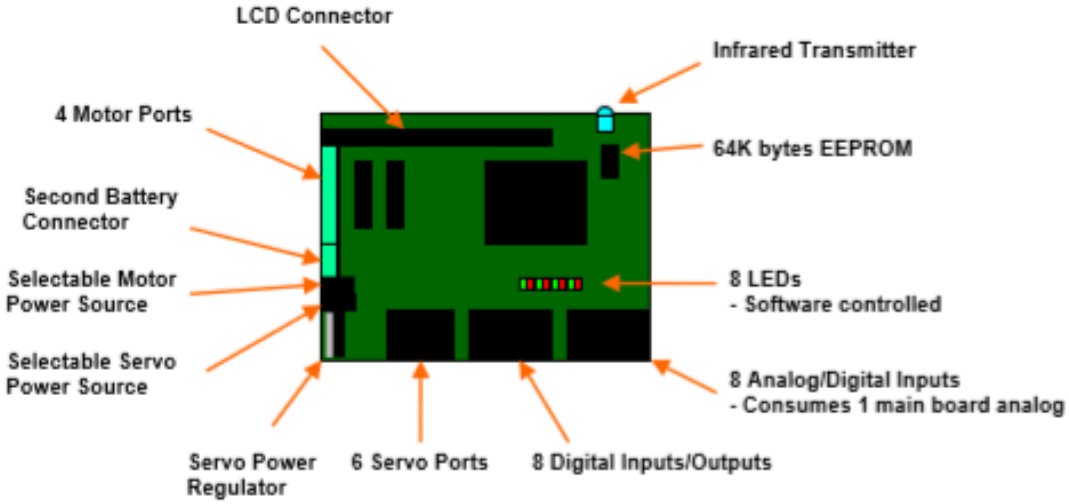
3.5 ELEKTRONIKA

Upravljačka ploča mobilnog robota se sastoji od dvije tiskane ploče. Prva tiskana ploča se zove engl. naziv „main board“ dimenzije 8,1 cm x 7,11 cm koja se sastoji od spojenih elemenata kao što su LCD ekran dimenzije 3,6 cm x 7,9 cm , LED diode, dugme za paljenje i gašenje robota, upravljačkog kontrolera i dr. elemenata (svi elementi glavnog tiskane ploče su prikazani na slici dolje).



Slika 3.28 Prikaz elemenata glavne tiskane ploče mobilnog robota

Druga tiskana ploča je ploča za proširenje engl. naziva „Expansion Board“ dimenzija 5,6 cm x 7,1 cm. Takva ploča za proširenja se montira na glavnu tiskanu ploču pomoću 40 pinskog priključka za proširenje i 14 pinskog priključka za LCD ekran. Elementi na takvoj tiskanoj ploči su četiri priključka za motore, šest servo priključaka i drugi spojeni elementi koji su prikazani na slici 3.29.



Slika 3.29 Prikaz elemenata na ploči za proširenje

Glavna tiskana ploča mobilnog robota sadrži 23 priključaka koja omogućuju spajanje različitih senzora i drugih elemenata mobilnog robota. Dodavanjem ploče za proširenje se dobiva puno veći broj priključaka i to njih 48 za senzore i druge elemente mobilnih robota. Da bi se moglo spojiti senzore i motore na upravljački kontroler potrebni su ulazni i izlazni priključci. Skoro svaki priključak se sastoji od tri do četiri priključnih igala koja označuje uključenje, uzemljenje i puls signal koji može se sastojati od jednog do dva impulsa. Na ploči upravljačkog sklopa jedno uz drugo su postavljeni priključci (slika 3.8.). Svaki od tog priključka je označen oznakom na ploči kontrolera koji označava vrstu i broj priključaka. Na IntelliBrain-Bot upravljačkom kontroleru postoje mnogi priključci kao:

1. analogni priključci koriste analogno-digitalni pretvarač za očitavanje napona od 0 do 5 [V] i pretvaraju u cjeloviti broj između 0 do 1023 gdje broj 0 označava 0 [V] a broj 1023 označava 5 [V] . Oznaka takvog priključka je A1-A7.

2. digitalni priključci su ulazni ili izlazni Boolean signali odnosno uključen/isključen ili istina/laž. Kada je postavljen kao izlaz onda digitalni ulaz daje 0 [V] kada je isključen odnosno laž a ako daje 5 [V] onda je uključen odnosno istina. Ako je digitalni priključak postavljen kao ulaz onda vraća vrijednost laž kada je signal blizu nuli a ako vraća vrijednost istina onda je signal blizu 5 [V]. Oznaka takvog priključka je I01-I013.

3. servo priključci se direktno priključuju na servo motore koji su inače prvobitno konstruirani za korištenje unutar modela aviona a kako zbog lagane mase, jednostavne elektronike unutar njih, niskih cijena se sada najviše koriste u mobilnoj robotici, oznake takvog priključka su S1-S8

4. serijski priključci se direktno povezuju sa kompleksnijim senzorima kao što je GPS ili kamera uređaji, također, se može postaviti Bluetooth serijski adapter da se računalo i robot mogu komunicirati uz pomoć WI-FI, oznake takvog priključka su COM1 i COM2 gdje se COM1 koristi kada želimo prenijeti napisani kod u upravljački kontroler robota.
5. infracrveni transmiter koji omogućuje paljenje i gašenje signala pomoću infracrvenog prijamnika te se koristi za komunikaciju između dva robota pomoću signala ili za signalizaciju drugog robota da stane ili krene.

Slika 3.30 Prikaz upravljačkog kontrolera sa raznim priključnim iglama

Kod elektronike mobilnog robota pojam interdisciplinarnosti se ostvaruje uz pomoć motora, senzora i drugih spojenih mehaničkih elemenata kroz povezivanje s pripadajućim priključcima na tiskane ploče koji sadrži upravljački kontroler.

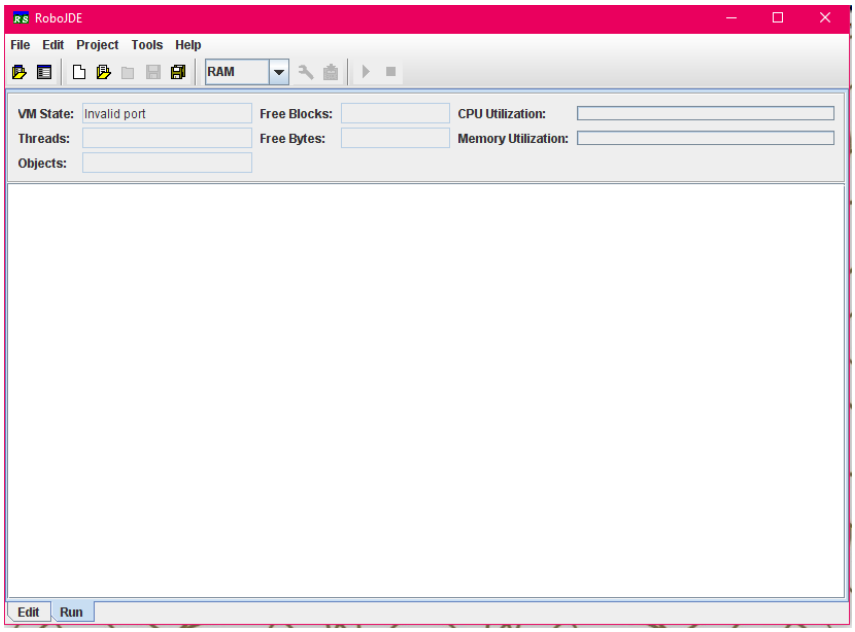
4. SUSTAV ZA PROGRAMIRANJE

IntelliBrain-Bot edukacijski mobilni robot tvrtke RidgeSoft koristi softversku programsku okolinu RoboJDE Java koja korisiti objektno orijentirani programski jezik Javu te takva programska okolina koristi se za edukaciju i male projekte u robotičkim djelatnostima radi lakšeg programiranja. Java omogućuje multi-threading odnosno mogućnost izvođenja više postupaka istovremeno ovisno o procesoru računala. RoboJDE omogućuje jednostavnu i brzu izradu programa koje izvode različite funkcije robota (slika 4.1). Unutar takve programske okoline imamo podjelu na tri velike cjeline a to su :

1) RoboJDE grafičko sučelje koje pruža jednostavno i brzo korištenje alata za pisanje kodova, kompajliranja, učitavanja programa na upravljački kontroler robota. On se izvodi uz pomoć Windows programa koji se nalazi u skoro svakom računalu a grafičko sučelje komunicira s roboto odnosno upravljačkim kontrolerom uz pomoć serijskog priključka naziva RS232.

2) RoboJDE virtualni stroj koji izvršava programe napisane u programskom jeziku Java i pruža kontrolu memorije, olakšanu sikronizaciju programa. Također, omogućuje zaštitu protiv čestih progrešaka tako da uoči pogreške kao što su null pokazivači odnosno pokazivači koji ne pokazuju na ništa određeno, pretrpavanje na stogu, progreške prilikom povezivanja itd. Takve pogreške se jasno ističu nakon kompaliranja te programer ima jasni uvid u svoje pogreške te lakše može ih ispraviti.

3) RoboJDE biblioteka klasa uključuje standardnu Java klasu, klase koje sadržavaju kodove vezane za mobilne robote te klase vezane za specifične upravljačke kontrolere koje omogućuju dobar početak za razvitak umjetne inteligencije robota. Da bi se moglo pristupiti cjeloj biblioteki klasa potrebno je kliknuti ikonu plave knjige koje se nalati u RoboJDE grafičkom sučelju



Slika 4.1 Prikaz RoboJDE programa

Java kao programski jezik koji se koristi za izradu raznih robotičkih projekata nudi razne prednosti kao :

a) korištenje objektno orijentiranog programskog jezika

- radi lakšeg razumijevanja i lakšeg otklanjanja grešaka

- mogu se koristiti klase programa RoboJDE

- omogućuje projekt rastaviti na funkcionalne cjeline tako da svaki član tima može bez problema prevoditi i izvoditi svoj dio projekta

b) jednostavnost uklanjanja grešaka

- java program štiti od čestih grešaka kao što su null pokazivači, pretrpanost stoga itd.

- nakon prevođenja ako postoje greške u programu one se pojavljuju u posebnom prozoru gdje piše redak koji bi se trebalo ispraviti odnosno javlja grešku koja se nalazi oko tog retka također piše koji dio retka nije dobar da li je možda pravopisna ili logičke naravi

c) izrađen u značajkama operativnog sustava

- java standardizira mnogobrojne operativne funkcije sistema kao što su sinkronizacija te razvrstavanje memorije

d) biblioteke programa

- mogućnost korištenja biblioteka drugih članova programske okoline

- java omogućuje mehanizam zvan Javadoc koji služi za generiranje sveobuhvatne dokumentacije programskog sučelja u standardni format koji se može vidjeti koristeći Internet pretraživač

e) bez ograničeno korištenje javnih biblioteka, literature radi što lakšeg shvaćanja programiranja u java programskom jeziku

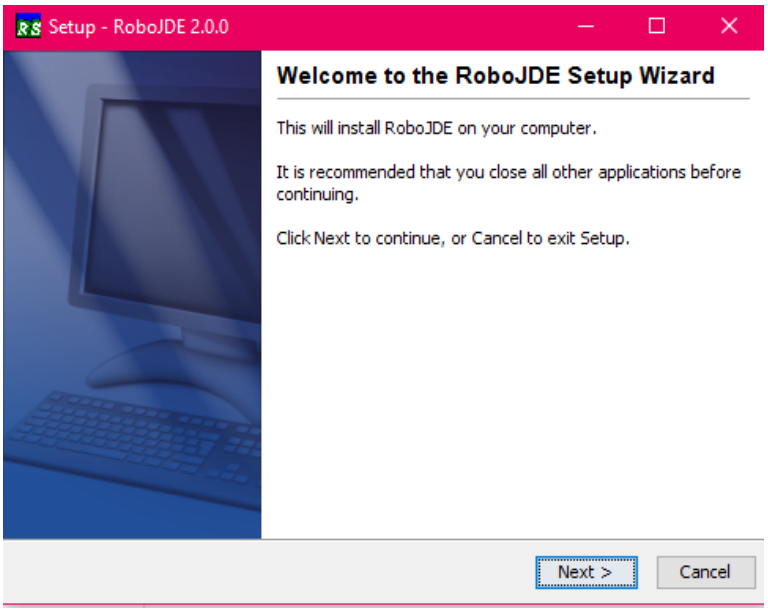
f) mogućnost prijenosa java programa na druge upravljačke kontrolore

g) omogućuje učenje modernih programskih vještina zbog mogućnosti korištenja na druge projekte itd.

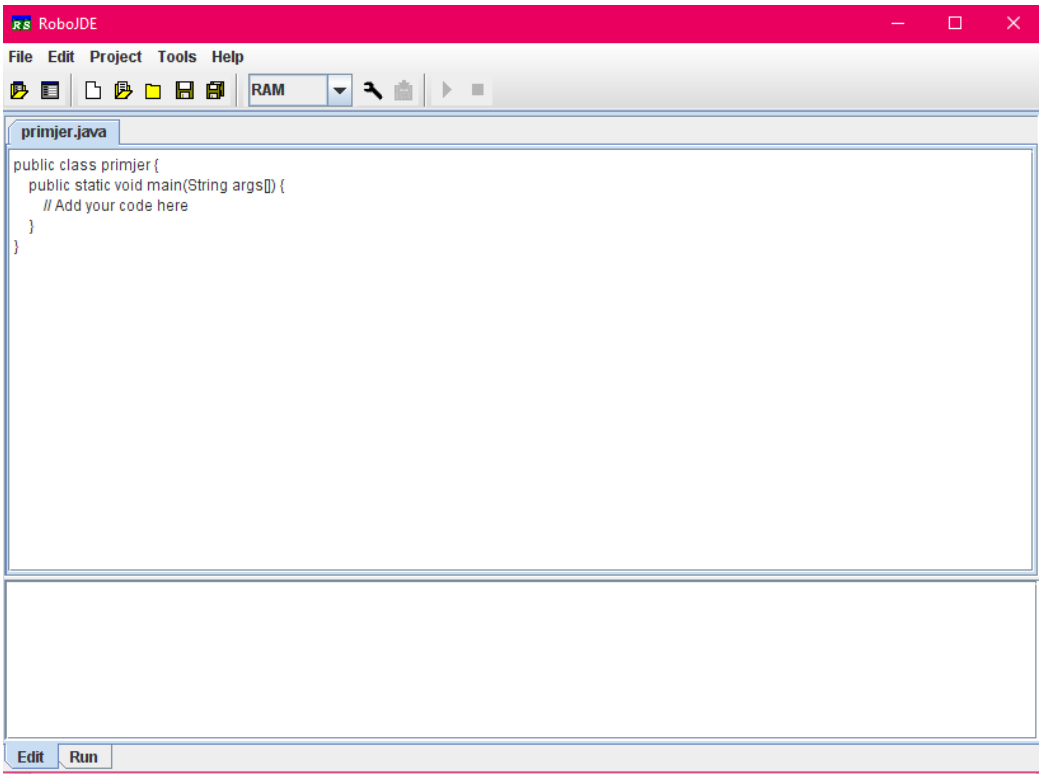
Program RoboJDE se nalazi na CD-ROM (slika 4.2.) ili na stranici tvrtke RidgeSoft gdje se lako može skinuti koristeći internet pretraživač. Instalacija samog programa je vrlo jednostavna samo je potrebno dva puta kliknuti na RoboJDESetup.exe datoteku te slijediti daljnje upute (slika 4.3.). Prije samog početka samostalnog rada potrebno je pregledati mapu naziva docs koji se nalazi na CDROM-u da bi se vidjelo već riješeni primjeri programa za RoboJDE program. Nakon instalacije programa na svom računalu potrebno je staviti licencu uz pomoć stranice programa jer bez nadogradnje licence neće biti moguća većina funkcija odnosno biblioteka koristeći samo RoboJDE-LITE licencu. Sa odgovarajućim ključem odnosno licence će se pružiti mogućnost potpunog funkcioniranja RoboJDE programa. Tada u RoboJDE programu moguće je kreirati novi projekt (slika 4.4), učitati neki napravljeni projekt ili jednostavno prevoditi napisani projekt te ga takvoga prenesti na upravljački sklop robota. Tijekom izrade projekta potrebno je pripaziti da se program rasporedi na smislene i

funkcionalne jedinice da ne bih došlo do pretrpavanja memorije te će tada program sporije prevoditi i možda se neće moći cijeli projekt staviti na upravljački kontroler robota koji ima 128 K FLASH memorije i 132 K RAM memorije. Program RoboJDE uključuje cjelovitu dokumentaciju za aplikaciju sučelja programiranja skraćeno API koja dopušta napisanom programu da se poveže sa za softverom IntelliBrain-Bot robota.

Slika 4.2 Prikaz RoboJDE CD-ROM



Slika 4.3. Prikaz nakon dva klika RoboJDESetup.exe datoteke

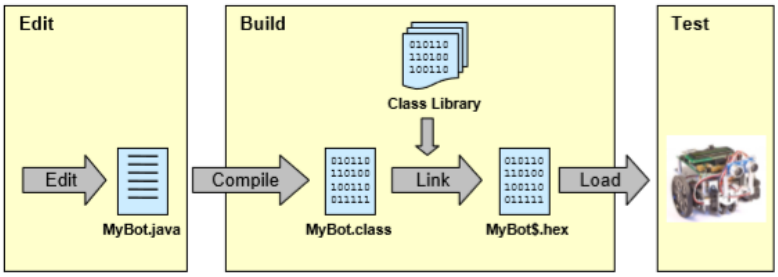


Slika 4.4. Prikaz novog kreiranog projekta za početak pisanja koda u RoboJDE programu

Prije početka pisanja koda u RoboJDE programu potrebno je napraviti konfiguraciju postavki programa na način da se za board Type postavi IntelliBrain, za serial port setting na ime PC serijskog porta kojeg mi odaberemo za povezivanje programa RoboJDE sa IntelliBrain-Bot robotom te se za baud rate postavi na vrijednost 38400 koji označuje brzinu prijenosa na IntelliBrain-Bot robotu. Svi navedeni elementi se nalaze na kartici settings u programu RoboJDE. Komunikacija između navedenog programa i upravljačkog kontrolera robota se postiže pomoću serijskog kabela koji je uključen u kit kompletu robota. Veza se uspostavlja na način da se muški kraj serijskog kabela postavlja na konektor naziva COM1 koji se nalazi na kontroleru robota a dok se ženski kraj kabela postavlja na COM port na računalu i odrediti koji port će se koristi od mogućih portova na računalu. Moguće je da računalo nema serijski port nego USB port potrebno je instalirati „USB na serijski adapter kabel“. Od adaptera se najviše koristi IOGear GUC232A (slika 4.5)

Slika 4.5 Adapter IOGear GUC232A

Program RoboJDE kao i svaki drugi programski jezik se sastoji od linija kodova tj. instrukcija kao što je Visual Basic, code::bloks i dr. imaju nekoliko faza izrade programa (slika 4.6)



Slika 4.6 Proces izrade programa za mobilnog robota IntelliBrain-Bot

Prilikom svakog puta programiranja programer će ponovno morati proći kroz sve faze izrade programa na način:

1. faza uređivanja (engl. „EDIT“) – sadrži pisanje, izmjena i brisanje koraka u programu unutar RoboJDE razvojne okoline.

2. faza stvaranja (engl. BUILD) – sadrži prevođenje (engl. compile) java dokumenta, povezivanje (engl. link) s bibliotekom gotovih klasa, preuzimanje programa iz programske okoline RoboJDE u upravljački kontroler mobilnog robota pritiskom na dugme „LOAD“ koje se nalazi na alatnoj traci navedene programske okoline. Prilikom prevođenja javlja se javni izvor prevoditelja nazvan „Jikes“ koji provjerava korake programa.

3. faza testiranja – gdje se primjenjuje napisani program na način da se pritisne dugme „RUN“ na alatnoj traci u RoboJDE programu ili pritiskom na dugme „START“ na upravljačkom kontroleru mobilnog robota

Prilikom faze stvaranja moguće su pogreške tijekom prevođenja nazvane engl. „compile errors“ napisane u posebnom prozoru unutar programa RoboJDE koje će zahtijevati povratak na fazu uređivanja gdje se koraci trebaju ispraviti te ponovno provjeriti zbog toga što izvršna datoteka ne može sadržati gramatičke greške te dvosmislenost. Također, može se pokazati da program sadrži više memorije nego što ima upravljački kontroler mobilnog robota. Isto tako, nakon uspješnog preuzimanja na kontroler robota postoji mogućnost u fazi testiranja da mobilni robot ne radi onako kako je programer očekivao te je potrebno se vratiti na početak tj. na fazu uređivanja programa.

Nakon svakog uspješnog dovršenog projekta stvara se jedna mapa koja sadrži četiri datoteke:

1. MyBot.java – java dokument koji sadrži napisane korake programa pod klasom imena MyBot

2. MyBot.class – dokument generiran nakon prevođenja koji sadrži java dokument sa klasama

3. MyBot$.hex – izvršna datoteka generirana od strane povezivača tj. engl. linker koja je povezala sve klase unutar programa

4. MyBot.rjp – datoteka koja sadrži svojstva projekta

Kako u programu RoboJDE se piše na „ljudskom“ jeziku tj. jeziku poznato ljudima to računalo ne razumije zato svaki programski jezik ima svoj asembler tj. niži simbolički jezik razumljiv računalu odnosno strojni jezik sačinjen od 0 i 1 . Asembler koji se koristi u navedenom programu ima svoju kraticu „ .S19“ u mapi svakog projekta programa.

Također, u programskom jeziku RoboJDE možemo pronaći više od 60 primjera koja pokazuju na koji način povezati različite elemente Java programa sa senzorima te upravljačkim kontrolerom robota.

Sustav za programiranje RoboJDE je područje informatike odnosno programski jezik koji sadrži naredbe , blokove naredbi, for i while petlje, if / else ili else if uvjeti i dr. koji se povezuje s ostatkom hardvera odnosno vidljivi dijelovi mobilnog robota i na takav način se ostvaruje pojam interdisciplinarnosti koji čini jedinstvenu cjelinu. Napisani kod u programskom jeziku RoboJDE sa sebe nema efekta tj. kod sam nema što pokrenuti ako ne postoji fizički napravljeni robot i tako se dolazi do navedenog pojma interdisciplinarnosti kod mobilnog robota IntelliBrain-Bot.

5. OSNOVNO O NAVIGACIJI MOBILNIH ROBOTA

Prilikom programiranja navigacije za mobilni robot potrebno je razmisliti na koji način riješiti problem navigacije. Jedino logično rješenje je da se problem podijeli na manje sastavnice. Kako bi se robot mogao što učinkovitije kretati potrebno je:

1. znati gdje želi ići
2. znati gdje se trenutno nalazi i koji smjer trenutno gleda
3. odrediti smjer do planiranog odredišta
4. krenuti i održavati smjer prema planiranom odredištu
5. stati nakon dolaska na odredište

Točna znanja o poziciji kotača je temeljni problem u mobilnoj robotici. Prilikom traženja rješenja su inženjeri, istraživači osmislili raznolike sustave, senzore i metode za pozicioniranje mobilnih robota.

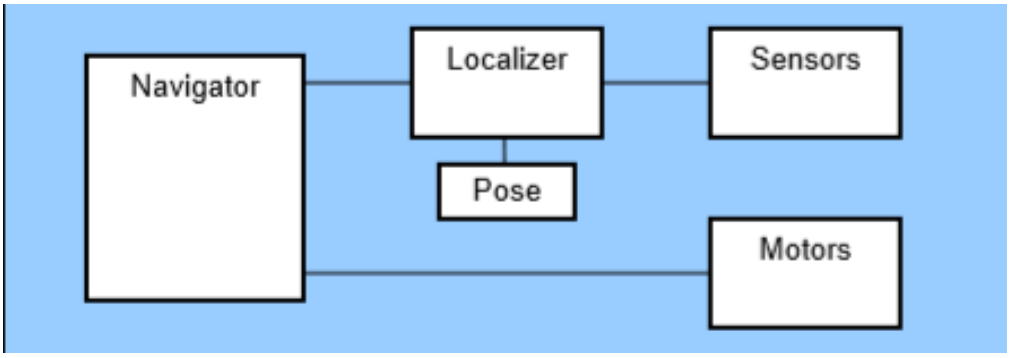
Navigatoru je znati gdje želi ići najjednostavnije jer je to odgovornost softvera više razine koji će specificirati tijek navigacije (slika 5.1). Na njemu je osigurati sredstvo za raspoznavanje koji će reći robotu gdje daje mora ići. Stoga će morati osigurati metode koje omogućuju postavljanja sljedećeg cilja. Postoje četiri metode koje služe navedenoj svrsi a to su:

a) moveTo – pomicanje na određenu poziciju

b) turnTo - okrenuti se u određenu stranu

c) go – kontinuirano pomicanje u jednom smjeru

d) stop



Slika 5.1 Dijagram klasa za navigaciju i lokalizaciju

Uz pomoć shaft enkodera i klase naziva „OdometricLocalizer class“ mobilni robot će moći znati gdje se sada nalazi i u kojem pravcu trenutno gleda. Koristeći navedenu klasu navigator će moći koristiti trigonometriju radi računanja pravca za sljedeće odredište. Slika 5.2 pokazuje kako robot koristi trigonometriju da bi došao do odredišta u ovom slučaju točke na kartezijevom koordinatnom sustavu. Na način, da je robot postavljen u točku (0,0) a

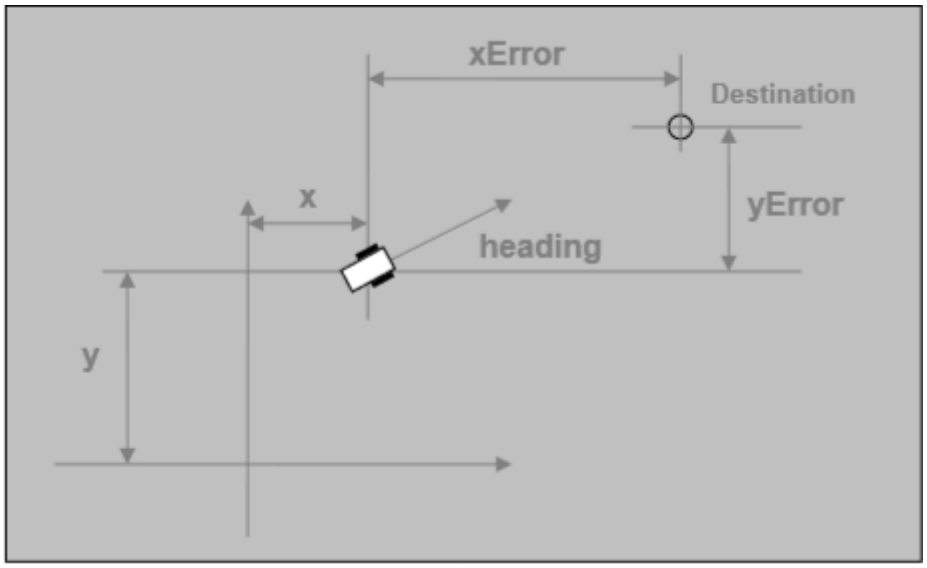
odredište ima točke po apcisi vrijednost xError a ordinata vrijednost yError dobivene na sljedeći način:

$$xError=destinationX-x$$

$$yError=destinationY-y$$

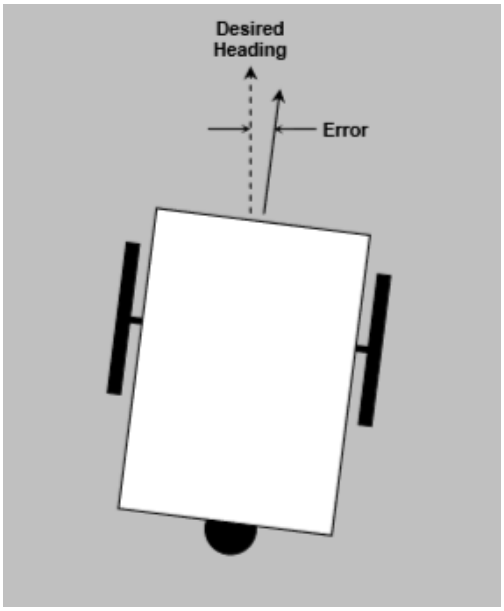
Pravac do odredišta zatvara određeni kut. Uz pomoć trigonometrije se dobiva pravac koristeći inverznu funkciju tangesa izračunom:

$$pravac= \arctan(yError/ xError)$$



Slika 5.2 Navigacija mobilnog robota prema točki odredišta

Jednom kada navigator izračuna smjer pravca mora primijeniti snagu motora da upravlja mobilnim robotom zbog održavanja pravca i kretnje prema naprijed. Navigator će postaviti istu snagu na oba dva kotača. Što bude više prijeđenog puta po pravcu navigatoru će biti teško se vratiti na pravi smjer pravca prema odredištu jer će se povećati razlika snaga (slika 5.3) . Ova tehnika je primjena metode proporcionalne kontrole koji se koristi kod kontroliranih dinamičkih sustava.



Slika 5.3 Prikaz pogreške u smjeru kretanja robota u točku odredišta

Kada mobilni robot stigne na približnu točku odredišta zbog pojavljivanja pogrešaka potrebno se je zaustaviti. Kod navigacije mobilnih robota potrebno je naglasiti da vrijednosti udaljenosti, točke odredišta neće biti precizne nego će biti oko tih vrijednosti jer se u većini slučajeva u robotici to prihvaća. Također, ciljati na točnu preciznu vrijednost se može ostvariti uz pomoć Pitagorinog teorema ali će trebati puno vremena za izračunati. Postoje mnogo izvora pogrešaka za navigaciju a to su kalibracija, klizanje kotača robota te metode za lokalizaciju . Pojam kalibracije se povezuje sa preciznošću promjera kotača i mjerene razlike osovina kotača pod uvjetom da lokalizator utječe na njegovu točnost. Dok je klizanje kotača rezultat pogrešaka lokalizacije.

Rješenja za navigaciju se mogu podijeliti na dvije glavne skupine. Razvojni programer mobilnih robota većinom kombinira dvije metode koje se uzimaju po jedna iz svake glavne skupine zbog nedostatka boljih metoda. Podjela glavnih skupina rješenja i njihove metode je ovakva:

- 1. Relativna pozicijska mjerenja : a) Odometrija

  b) Inercijska navigacija
- 2. Apsolutna pozicijska mjerenja : a) Metoda magnetičnih kompasa

  b) Metoda „active beacons“

  c) GPS

  d) Metoda pozicioniranja putem aktivnih orijentira

  e) Metoda pozicioniranja putem prirodnih orijentira

  e) Metoda korištenja karte u navigaciji

U daljnjim potpoglavljima će biti opisani samo četiri metode navigacije vezane za odometriju, navigaciju putem aktivnih orijentira, navigaciju uz pomoć prirodnih orijentira te metodu korištenja karte u navigaciji odnosno navigacija temeljena na video signalu.

5.1 ODOMETRIJA

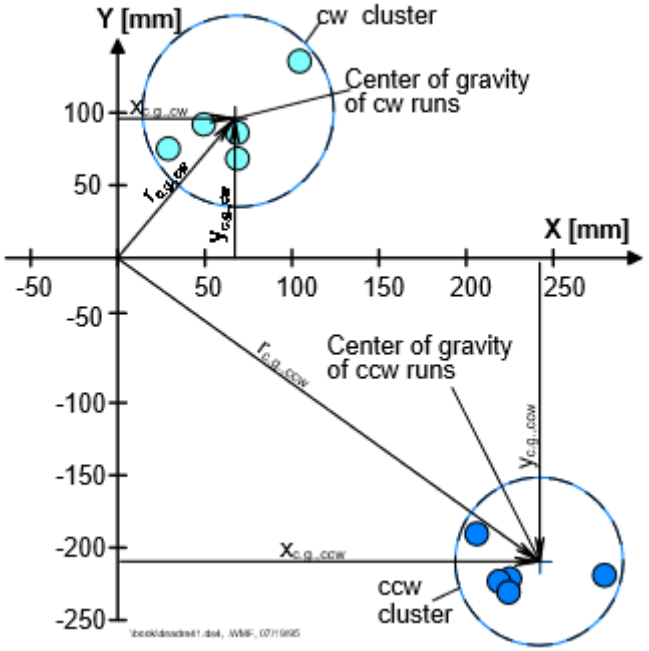
Metoda navigacije kod pozicioniranja mobilnog robota koja se široko primjenjuje i omogućuje dobru kratkoročnu preciznost. Također, često se koristi zbog manje cijene primjene i isto tako omogućuje visoke stope odabira uzroka. Temeljna ideja odometrije je uklapanje inkrementalnih informacija tijekom vremena što slijedi neizbježno nakupljanje pogrešaka. Postoje orijentacijske pogreške koje će prouzrokovati velike bočne pozicijske pogreške i povećavati će se proporcionalno sa prijeđenim putem mobilnog robota. Unatoč ovim navedenim pogreškama mnogi istraživači se slažu da je metoda odometrije jedna bitna značajka u navigacijskim sustavima i da se zadaci navigacije mogu jednostavno riješiti ako se samo poboljša preciznost. Odometrija je zasnovana na jednostavnim jednadžbama koje će biti istinite samo onda kada okretanje kotača mogu biti precizno prevesti u linearni pomak u odnosu na pod.

U slučaju klizanja kotača robota i drugih uzroka okretanje kotača možda neće biti proporcionalno prenesene u linearni pomak. Rezultat toga su pogreške koje se dijele na sistemske pogreške i pogreške koje nisu sistemske. Pod sistemske pogreške smatramo rezultate kinematske nesavršenosti mobilnog robota. Primjeri takvih pogrešaka su nejednaki promjeri kotača, nesigurnost u vezi razmaka osovina. Pod pogreške koje nisu sistemske se smatraju pogreške koje se javljaju kao rezultat komunikacije između poda i kotača robota. Klizanje kotača, sudaranja te razbijanja su primjeri pogrešaka koje nisu sistemske. Robotičari Borenstein i Feng su izveli kalibracijski test radi smanjenja sistemskih pogrešaka tijekom odometrijske navigacije na mobilnom robotu diferencijalnog sustava upravljanja. Kalibracijski test je osnovan na UMBmark testu . U navedenom postupku se UMBmark test ponavlja pet puta u svakom smjeru nazvanim „ cw“ i „ccw“ kako bi pronašli točke $X_{c.g.cw}$ i $X_{c.g.ccw}$. Uz pomoć navedenih točaka se mogu dobiti kalibracijske konstante koje će se naći u osnovnom odometrijskom proračunu. Kalibracijske konstante se dobivaju na način:

$$r_{c.g.,cw} = \sqrt{(x_{c.g.,cw})^2 + (y_{c.g.,cw})^2}$$

$$r_{c.g.,ccw} = \sqrt{(x_{c.g.,ccw})^2 + (y_{c.g.,ccw})^2} .$$

Tipični rezultati nakon kalibracijskog testa (slika 5.4) je oko 330 mm za kalibracijskog TRC¨s LabMate robot a za robote koji nisu kalibracijski je rezultat 24 mm.



Slika 5.4 Prikaz rezultat nakon kalibracijskog testa

Robotičar Borenstain je 1995. godine razvio metodu za otkrivanje i rješavanja odbijajući odometrijske pogreške koje nisu sistemske u sustavu mobilnog robota. Ovom metodom se dvije suradničke platforme mogu kontinuirano i višestruko ispravljati pogreške koje nisu sistemske ali i neke sistemske pogreške. Robot OmniMate sadrži ovakvu metodu i gotovo je zaštićen od kvrga, pukotina i drugih mogućih oštećenja na podu (slika 5.5).

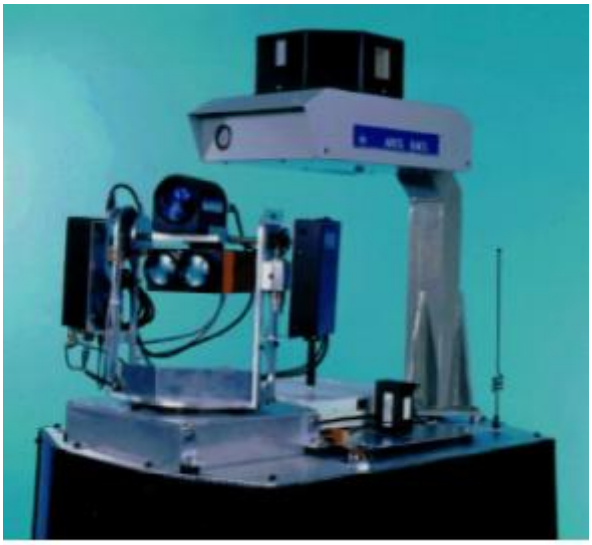
Slika 5.5 OmniMate komercijalni robot

Interdisciplinarnost kod mobilnih robota vezano za odometrijsku navigaciju se ostvaruje povezivanjem napisanog koda i kotača robota odnosno povezivanjem disciplina programiranja i matematike. Disciplina programiranja za odometrijsku navigaciju uključuje vlastite klase, metode i funkcije za motore, razne senzore, LCD ekran i dr. mehaničkih elemenata mobilnog robota. Kod discipline matematike za odometrijsku navigaciju uključuje računanje lokalicije mobilnog robota sa varijeblama i dr. matematičke formule vezano za kretanje mehaničkog elementa kotača mobilnog robota.

5.2 METODA NAVIGACIJE UZ POMOĆ ORIJENTIRA

Orijentiri su posebne mogućnosti koje robot može prepoznati iz senzornih ulaza. Također, moraju biti dostupni u radnom okruženju robota. Poprimaju geometrijske oblike npr. linije, krugove te pravokutnike i mogu uključivati dopunske informacije npr. obrazac za bar kod. Uobičajeno, orijentiri imaju fiksnu i poznatu poziciju u vezi su sa robotom koji može se sam lokalizirati . Oni su pažljivo odabrani da bi se mogli lako prepoznati. Na primjer, podloga mora biti dovoljno kontrastna. Prije nego što mobilni robot počne koristiti orijentire za navigaciju karakteristike orijentira moraju biti već poznati i spremljeni u memoriju robota. Glavna zadaća u navigaciji da prepoznaje pouzdani orijentir i izračunati trenutnu poziciju mobilnog robota.

Da bi se pojednostavio problem stjecanja orijentira pretpostavlja se da trenutna pozicija robota i orijentacija približno poznata tako da mobilni robot traži orijentire u ograničenom području. Stoga je dobra preciznost odometrije preduvjet za uspješan pronalazak orijentira. Postoje aktivni i prirodni tipovi orijentira. Definicija prirodnih orijentira je da su to objekti ili značajke koje se već nalaze u okruženju i imaju funkciju koja je različita od funkcije navigacije za mobilni robot. Glavni problem u navigaciji prirodnog orijentira je pronalazak i spajanje značajnih karakteristika senzora.Ova metoda omogućuje prilagodljivost i ne zahtjeva preinake oko okruženja. Primjer robota za navigaciju prirodnog orijentira je AECL¨s ARK Project (slika dolje).



Slika 5.6 ARK¨ s ARK Project koji koristi navigaciju prirodnog orijentira

Pomoću aktivnih orijentira otkrivanja su puno lakša koja su dizajnirana za najpovoljniji kontrast. Točne dimenzije i oblici aktivnih orijentira su poznata već unaprijed. Većina sustava za navigaciju pomoću aktivnih orijentira su bazirana na računalnim vidom. Primjer robota za navedenu navigaciju je MDARS koji se koristi u vojsci (slika 5.7)

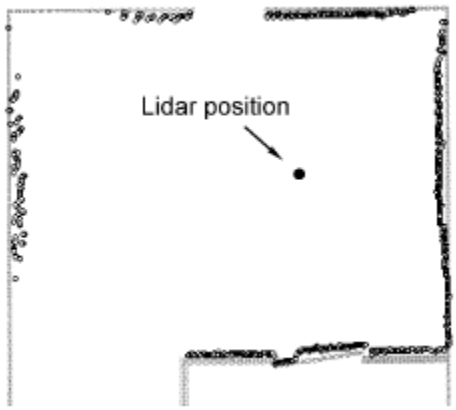
Slika 5.7 Prikaz robota za sustav procjene mobilnih detekcija i sustava odgovora

Tijekom metode navigacije pomoću orijentira kod mobilnih robota se ostvaruje pojam interdisciplinarnosti koje povezuje discipline kao što su matematika,elektronika i informatika. Interdisciplinarnost očituje senzorne ulaze mobilnog robota koji prepoznaju zadane elemente u radnom okruženju robota, geometrijske oblike koje orijentir poprima te memoriju robota što sadrži poznate orijentire.

5.3. NAVIGACIJA TEMELJENA NA VIDEO SIGNALU

Metoda korištenja karata odnosno navigacija temeljena na video signalu je tehnologija u kojoj robot koristi senzore radi stvaranja svojeg lokalnog područja. Karta koja se dobiva uz pomoć robota se uspoređuje sa globalnom kartom koja je već prije spremljena u memorji robota. Ako je usporedba karata točna onda robot može odrediti svoju trenutnu poziciju i orijentaciju unutar snimljenog područja. Globalna karta koja je već spremljena u memoriju uređaja može biti tipa CAD modela područja ili ranijeg snimanog područja. Prednosti navedene metode navigacije su korištenje prirodne strukturne događaja u tipičnim unutarnjim područjima koji izvlače informacije o poziciji bez izmjene područja te sa ponekim novim algoritmima omogućuje robotu da nauči novo okruženje i da poboljša preciznost pozicioniranja tijekom korištenja navedene navigacije.

Nedostatci ove metode su strogi zahtjevi za točnost karata dobivene preko senzora, zahtjevi za dovoljno statičnih, jednostavnih značajki koje se može razlikovati kada bi se koristile za uspoređivanje karata. Navigacija pomoću karata odnosno pozicioniranja u području je ograničena zbog mogućnosti korištenja samo jednostavnijih okruženja. Proces navigacije započinje sa samostanim istraživanjem robota da označi novo trenutno područje . Pretpostavlja se da robot kreće u istraživanje bez ikakvih znanja o trenutnom području. Zatim, u određenoj strategiji pokreta slijedi usmjereno povećanje količine označenih područja u najmanjem vremenu. Takva strategija strogo ovisi o kojem tipu senzora se koristi za istraživanje područja. Primjer početka navigacije je na slici 5.8.



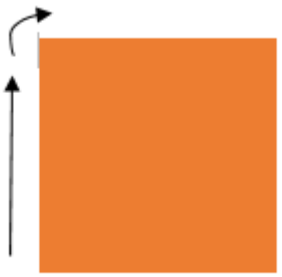
Slika 5.8 Primjer početka navigacije pomoću video signala

Kasnije nakon snimanja područja dolazi do usporedbe područja trenutnog snimljenog i već pohranjene karta u memoriji ili prijašnjeg snimanja. Usporedba karata je uspješna već kada prepozna prve zajedničke značajke. Kod svake metode navigacije pa tako i metoda navigacije temeljena na video signalu ostvaruje pojam interdisciplinarnosti koji povezuje discipline matematike i programiranja na način povezivanja prepoznavanja slike. Također, navedni pojam povezuje i disciplinu elektronike radi puta signala kroz mobilni robot te povezuje mehaničke elemente kao što su motori,senzori, upravljački kontroleri i dr.

6. PRIMJER PROGRAMIRANJA KROZ POVEZANE ZADATKE

Interdisciplinarnost kod mobilnog robota IntelliBrain-Bot se očituje kao jedinstvena cjelina u kojoj se povezuju discipline matematika i programiranja te se također očituje povezanost elektroničkog dijela odnosno upravljačkog kontrolera mobilnog robota i mehaničkih dijelova kao što su motori, razni senzori i dr. Programi koji će postati dio ovog poglavlja pokazati će učenje programiranja sa povezivanjem znanja iz dijela mehanike i elektronike mobilnog robota koristeći osnovne matematičke funkcije.

U ovo poglavlju će se objasniti na koji način se može iz jednog osnovog zadatka modificirati ostali povezani zadatci. Osnovni zadatak je kretanje robota u geometrijskom obliku kvadrata (slika 6.1.). Na toj slici se točno pokazuju kretnje robota koji će se ostvariti prilikom povezivanja s principom rada programa napisanog u prilogu 1. U napisanom programu se prvo provlače klase povezane s kontrolerom naziva IntelliBrain i servo motora naziva Servo nakončega je moguće upravljati servo motorima robotima i to je način ostvarivanja pojma interdisciplinarnosti kod mobilnog robota.



Slika 6.1. Kretanje mobilnog robota u obliku kvadrata sa for petljom

Vožnja robota je programirana na način da će se 4 puta provesti ista for petlja u kojoj će se robot prvo kretati ravno određeno vrijeme od dvije sekunde koje odgovara dijelu koda od 2000 [ms]. Nakon čega će se robot zakrenuti za približno 90 stupnjeva na način da se motori vrte u suprotnim smjerovima određeni vremenski interval od 0,6 sekundi koje odgovara dijelu koda napisanog 600 [ms]. Kad se for petlja izvrti isti manevar mobilni robot bi se trebao vratiti na početnu točku i tako iscrtati svojim kretanjima geometrijski oblik kvadrata. Dio koda koji se prethodno objasnio se u programu RoboJDE piše ovako:

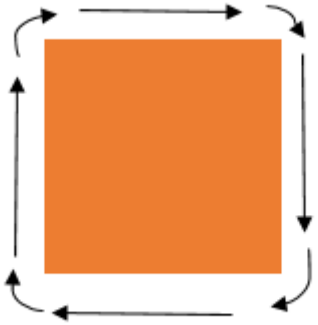
```
for (int i = 0; i < 4; ++i) {
        // drive forward
        leftServo.setPosition(100);
        rightServo.setPosition(0);
        Thread.sleep(2000);
```

```
        // rotate clockwise approximately 90 degrees
        leftServo.setPosition(100);
        rightServo.setPosition(100);
        Thread.sleep(600);
    }
```

Dio koda za vožnju robota u obliku kvadrata se može napisati na način gdje nema for petlje koja će ponavljati kretanje prema naprijed i zakretanje robota za približno 90 stupnjeva nego jednostavno će se cijeli proces vožnje robota napisati postepeno u kodu. Takav napisani kod je nepregledan i zauzima više memorije te će biti potrebno više vremena da se program preuzme na upravljački kontroler robota. Vožnja mobilnog robota u obliku kvadrata bez ponavljajuće for petlje (slika 6.2.) glasi :

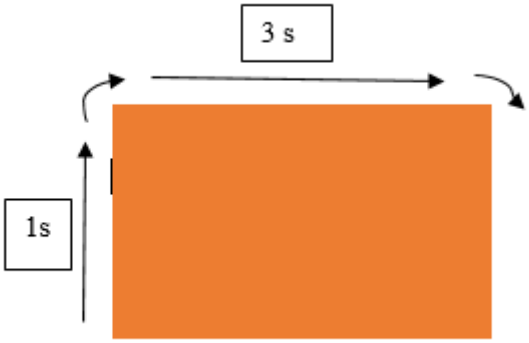
```
        // drive forward
        leftServo.setPosition(100);
        rightServo.setPosition(0);
        Thread.sleep(2000);
        // rotate clockwise approximately 90 degrees
        leftServo.setPosition(100);
        rightServo.setPosition(100);
        Thread.sleep(600);
        // drive forward
        leftServo.setPosition(100);
        rightServo.setPosition(0);
        Thread.sleep(2000);
        // rotate clockwise approximately 90 degrees
        leftServo.setPosition(100);
        rightServo.setPosition(100);
        Thread.sleep(600);
        // drive forward
        leftServo.setPosition(100);
        rightServo.setPosition(0);
        Thread.sleep(2000);
        // rotate clockwise approximately 90 degrees
        leftServo.setPosition(100);
```

```
rightServo.setPosition(100);
Thread.sleep(600);
// drive forward
leftServo.setPosition(100);
rightServo.setPosition(0);
Thread.sleep(2000);
// rotate clockwise approximately 90 degrees
leftServo.setPosition(100);
rightServo.setPosition(100);
Thread.sleep(600);
```



Slika 6.2. Vožnja robota po putanji bez ponavljajućeg elementa

Gore prikazani dio koda kretnje mobilnog robota u obliku kvadrata sa ponavljajućem elementom se može modificirati za rješenje zadatka kretnje robota po putanji geometrijskog oblika pravokutnika omjera stranica 1: 3. (slika 6.3).



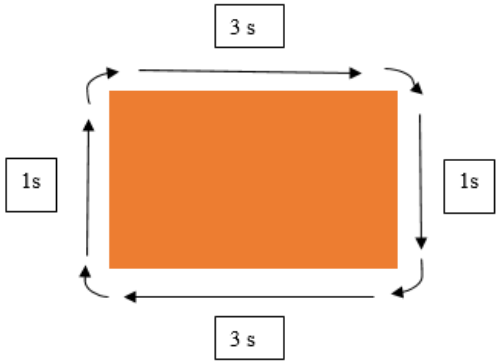
Slika 6.3. Kretanje mobilnog robota u obliku pravoktnika sa for petljom

Na način da se programira vožnja robota koja će imati for petlju. Takva petlja će se provesti dva puta u kojoj će robot voziti ravno približno jednu sekundu, zakretati se za približno 90 stupnjeva, voziti ravno tri sekunde te ponovno se zakrenuti za približno 90 stupnjeva. Odnosno vožnja robota u obliku pravokutnika je modificirana u odnosu na vožnju u obliku kvadrata na način da ima dvije ravne vožnje i dva zakretanja za približno 90 stupnjeva. Takav prethodno objašnjen dio koda za vožnju robota u obliku pravokutnika se programira ovako:

```
for (int i = 0; i < 2; ++i) {
// Vozi ravno jednu sekundu
        leftServo.setPosition(100);
        rightServo.setPosition(0);
        Thread.sleep(1000);
// zakrece se za priblizno 90 stupnjeva
        leftServo.setPosition(100);
        rightServo.setPosition(100);
        Thread.sleep(625);
// Vozi ravno tri sekunde
        leftServo.setPosition(100);
        rightServo.setPosition(0);
        Thread.sleep(3000);
// zakrece se za priblizno 90 stupnjeva
        leftServo.setPosition(100);
        rightServo.setPosition(100);
        Thread.sleep(625);
}
```

Gore navedeni dio koda se nalazi u prilogu 2. gdje se nalazi cijeli program za vožnju mobilnog robota u obliku pravokutnika. Cjeloviti program ostvaruje pojam interdisciplinarnosti kod mobilnog robota jer programiranjem vožnje robota se povezuje sa upravljačkim kontrolerom i servo motorima koji omogućuju kretanje kotača mehaničog dijela robota.

Kao i u prethodnom zadatku gdje je bio oblik kvadrata tako i kod kretanja mobilnog robota u obliku kvadrata se može modificirati rješenje zadatka bez korištenja for petlje (slika 6.4). Takav kod je više nepregledniji jer ima četiri ravne vožnje i četiri zakretanja za približno 90 stupnjeva.



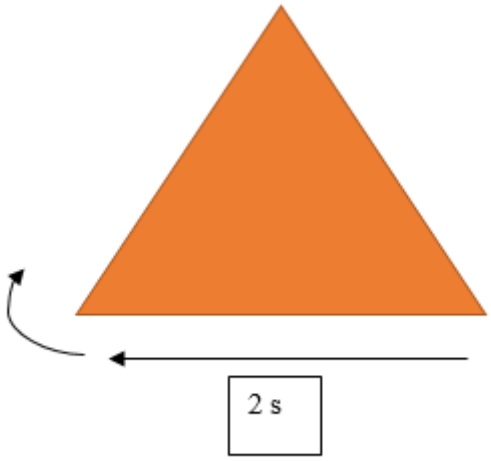
Slika 6.4. Kretanje mobilnog robota u obliku pravokutnika bez korištenja for petlje

Takav kod može se definirati ovako:

```
// Vozi ravno jednu sekundu
        leftServo.setPosition(100);
        rightServo.setPosition(0);
        Thread.sleep(1000);
// zakrece se za priblizno 90 stupnjeva
        leftServo.setPosition(100);
        rightServo.setPosition(100);
        Thread.sleep(625);
// Vozi ravno tri sekunde
        leftServo.setPosition(100);
        rightServo.setPosition(0);
        Thread.sleep(3000);
// zakrece se za priblizno 90 stupnjeva
        leftServo.setPosition(100);
        rightServo.setPosition(100);
        Thread.sleep(625);
// Vozi ravno jednu sekundu
        leftServo.setPosition(100);
        rightServo.setPosition(0);
        Thread.sleep(1000);
// zakrece se za priblizno 90 stupnjeva
```

```
            leftServo.setPosition(100);
            rightServo.setPosition(100);
            Thread.sleep(625);
        // Vozi ravno tri sekunde
            leftServo.setPosition(100);
            rightServo.setPosition(0);
            Thread.sleep(3000);
        // zakrece se za priblizno 90 stupnjeva
            leftServo.setPosition(100);
            rightServo.setPosition(100);
            Thread.sleep(625);
```

Rješenje za zadatak sa vožnjom mobilnog robota u obliku pravokutnika sa ponavljajućom for petljom se može modificirati za rješenje zadatka kretanje mobilnog robota u obliku jednakostraničnog trokuta (slika 6.5).



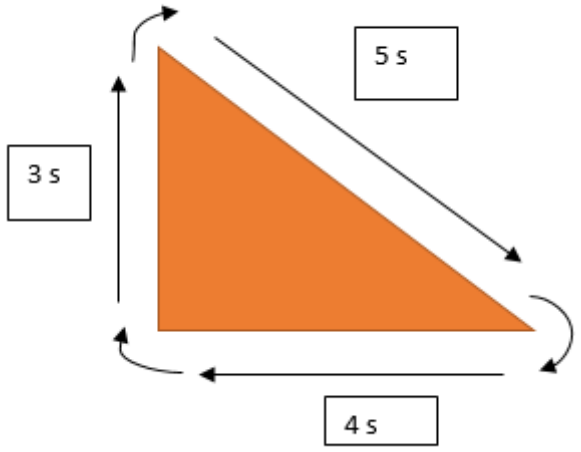
Slika 6.5. Vožnja mobilnog robota u obliku jednakostraničnog trokuta sa ponavljajućom for petljom

Na način da se u for petlji mobilni robot vozi ravno dvije sekunde te se zakreće za približno 60 stupnjeva umjesto 90 stupnjeva kod vožnje mobilnog robota u obliku pravokutnika. Takav modificirani kod se može definirati na način:

```
for (int i = 0; i < 3; ++i) {
        // Vozi ravno dvije sekunde
            leftServo.setPosition(100);
            rightServo.setPosition(0);
            Thread.sleep(2000);
        // zakrece se za priblizno 60 stupnjeva
            leftServo.setPosition(100);
            rightServo.setPosition(100);
            Thread.sleep(840);
        }
```

Gore navedeni dio koda koji služi za rješenje modifacije za kretanje mobilnog robota u obliku jednakostraničnog trokuta se nalazi u cjelovitom napisanom kodu za takvo kretanje u prilogu 3. Također, programiranjem programa se povezuju servo motori zaslužni za kretanje kotača i upravljački kontroler te se ostvaruje interdisciplinaranosti kod mobilnog robota.

Također, rješenje prethodnog zadatka vožnje robota u obliku jednakostraničnog trokuta se može koristiti i za dobivanje rješenja zadatka vožnje mobilnog robota u obliku pravokutnog trokuta ali sa potrebnim modifikacijama (slika 6.6.)



Slika 6.6 Vožnja mobilnog robota u obliku pravokutnog trokuta sa omjerom stranica 3:4:5

Modifikacije koje su napravaljene za riješenje gore navedenog zadatka je da mobilni robot vozi ravno 3 sekunde pa se zakreće za približno 60 stupnjeva onda ponovno vozi ravno 5 sekundi pa se zakreće približno 30 stupnjeva te ponovno ravno vozi približno 4 sekunde i na

kraju se zakreće za približno 90 stupnjeva. Ovako objašnjene modifikacije rješenja za zadatak se pišu u dijelu koda dolje prikazano.

```
// Vozi ravno 3 sekunde
        leftServo.setPosition(100);
        rightServo.setPosition(0);
        Thread.sleep(3000);
// zakrece se za priblizno 60 stupnjeva
        leftServo.setPosition(100);
        rightServo.setPosition(100);
        Thread.sleep(900);
// Vozi ravno 5 sekundi
        leftServo.setPosition(100);
        rightServo.setPosition(0);
        Thread.sleep(5500);
// zakrece se za priblizno 30 stupnjeva
        leftServo.setPosition(100);
        rightServo.setPosition(100);
        Thread.sleep(1110);
// Vozi ravno 4 sekunde
        leftServo.setPosition(100);
        rightServo.setPosition(0);
        Thread.sleep(4600);
// zakrece se za priblizno 90 stupnjeva
        leftServo.setPosition(100);
        rightServo.setPosition(100);
        Thread.sleep(625);
```

Gore navedeno riješenje modifikacije za vožnju mobilnog robota se nalazi i u prilogu 4. gdje je cjelovito napisani kod za navedeni zadatak koji kao i ostala prethodna zadatka ostvaruju interdisciplinarnost mobilnog robota.

7. ZAKLJUČAK

Primjerom povezanih zadataka se ostvaruje interdisciplinarnost koji povezuje različite discipline kao što su matematika, informacijske znanosti, mehanike i elektronike robota. Također, zbog interdisciplinarnost sustava lako se modificiraju slični zadatci. Osim povezanih zadataka postoju razne metode navigacije koje koriste interdisciplinarnost koja se očituje u povezivanju mehaničkih i elektroničkih dijelova robota. Zbog njihove interdisciplinarnosti, mobilni roboti (poput mobilnog robota IntelliBrain-Bot koji je kao primjer u ovom završnom radu korišten za prikaz svih elemenata) mogu dobro služiti u obrazovanju korisnika da na što jednostavniji način uđu u svijet robotike i programiranja. Također, učenjem o mobilnim robotima odnosno programiranjem njihovog ponašanja u skladu s njihovim mogućnostima do izražaja dolazi interdisciplinarna povezanost elemenata mobilnog robota koji svi zajedno čine jedinstvni sustav.

LITERATURA

1. Preston, Scott : „The Definitive Guide to Building Java Robots“, Apress, New York (SAD), 2006.godina
2. Krstulović, Ante : „Uvod u industrijsku robotiku“, Hrvatska zajednica tehničke kulture, Zagreb, 2000.godina
3. Tvrtka RidgeSoft : „RoboJDE User Guide“, s interneta, http://www.ridgesoft.com/, 10.11.2017
4. Tvrtka RidgeSoft: „Creating your first IntelliBrain program“, s interneta, http://www.ridgesoft.com/, 10.11.2017.god.
5. Tvrtka RidgeSoft: „Programming your robot to perform basic maneuvers“, s interneta, http://www.ridgesoft.com/ , 10.11.2017.god.
6. Tvrtka RidgeSoft: „Creating a user interface for your robot“, s interneta, http://www.ridgesoft.com/ , 10.11.2017.god
7. Tvrtka RidgeSoft: „Creating shaft encoders foe wheel position sensing“, s interneta, http://www.ridgesoft.com/ , 10.11.2017.god
8. Tvrtka RidgeSoft: „Enabling your robot to keep track of its position“, s interneta, http://www.ridgesoft.com/ , 10.11.2017.god
9. Tvrtka RidgeSoft: „Programming your robot to navigate“, s interneta, http://www.ridgesoft.com/ , 10.11.2017.god
10. Tvrtka RidgeSoft: „Exploring robotics with the IntelliBrain-Bot“, s interneta, http://www.ridgesoft.com/ , 10.11.2017.god

POPIS SLIKA

PRILOG 1

```
import com.ridgesoft.io.Display;
import com.ridgesoft.intellibrain.IntelliBrain;
import com.ridgesoft.robotics.Servo;


public class MyBot {
public static void main(String args[]) {
	try {
		Display display = IntelliBrain.getLcdDisplay();
			display.print(0, "Politehnika je");
			display.print(1, "zivot moj...");


	Servo leftServo = IntelliBrain.getServo(1);
	Servo rightServo = IntelliBrain.getServo(2);


		for (int i = 0; i < 4; ++i) {
		// drive forward
			leftServo.setPosition(100);
			rightServo.setPosition(0);
			Thread.sleep(1000);
		// rotate clockwise approximately 90 degrees
			leftServo.setPosition(100);
			rightServo.setPosition(100);
			Thread.sleep(625);
		}
			leftServo.off();
			rightServo.off();
	}

	catch (Throwable t) {
	t.printStackTrace();
}
}
}
```

PRILOG 2

```
import com.ridgesoft.io.Display;
import com.ridgesoft.intellibrain.IntelliBrain;
import com.ridgesoft.robotics.Servo;

public class MyBot {
public static void main(String args[]) {
	try {
		Display display = IntelliBrain.getLcdDisplay();
			display.print(0, "Politehnika je");
			display.print(1, "zivot moj...");

	Servo leftServo = IntelliBrain.getServo(1);
	Servo rightServo = IntelliBrain.getServo(2);

		for (int i = 0; i < 2; ++i) {
		// Vozi ravno jednu sekundu
			leftServo.setPosition(100);
			rightServo.setPosition(0);
			Thread.sleep(1000);
		// zakrece se za priblizno 90 stupnjeva
			leftServo.setPosition(100);
			rightServo.setPosition(100);
			Thread.sleep(625);
		// Vozi ravno tri sekunde
			leftServo.setPosition(100);
			rightServo.setPosition(0);
			Thread.sleep(3000);
		// zakrece se za priblizno 90 stupnjeva
			leftServo.setPosition(100);
			rightServo.setPosition(100);
			Thread.sleep(625);
		}
			leftServo.off();
			rightServo.off();
	}

	catch (Throwable t) {
	t.printStackTrace();
}
}
}
```

PRILOG 3

```
import com.ridgesoft.io.Display;
import com.ridgesoft.intellibrain.IntelliBrain;
import com.ridgesoft.robotics.Servo;
public class MyBot {
public static void main(String args[]) {
	try {
		Display display = IntelliBrain.getLcdDisplay();
			display.print(0, "ARIANA");
			display.print(1, "ADRIAN");
	Servo leftServo = IntelliBrain.getServo(1);
	Servo rightServo = IntelliBrain.getServo(2);
		for (int i = 0; i < 3; ++i) {
		// Vozi ravno dvije sekunde
			leftServo.setPosition(100);
			rightServo.setPosition(0);
			Thread.sleep(1000);
		// zakrece se za priblizno 60 stupnjeva
			leftServo.setPosition(100);
			rightServo.setPosition(100);
			Thread.sleep(840);
		}
			leftServo.off();
			rightServo.off();
	}

	catch (Throwable t) {
	t.printStackTrace();
}
}
}
```

PRILOG 4

```
import com.ridgesoft.io.Display;
import com.ridgesoft.intellibrain.IntelliBrain;
import com.ridgesoft.robotics.Servo;
public class MyBot {
public static void main(String args[]) {
	try {
		Display display = IntelliBrain.getLcdDisplay();
			display.print(0, "ARIANA");
			display.print(1, "ADRIAN");
	Servo leftServo = IntelliBrain.getServo(1);
	Servo rightServo = IntelliBrain.getServo(2);

		// Vozi ravno 3 sekunde
			leftServo.setPosition(100);
			rightServo.setPosition(0);
			Thread.sleep(3000);
		// zakrece se za priblizno 60 stupnjeva
			leftServo.setPosition(100);
			rightServo.setPosition(100);
			Thread.sleep(900);
		// Vozi ravno 5 sekundi
			leftServo.setPosition(100);
			rightServo.setPosition(0);
			Thread.sleep(5500);
		// zakrece se za priblizno 30 stupnjeva
			leftServo.setPosition(100);
			rightServo.setPosition(100);
			Thread.sleep(1110);
```

```
                // Vozi ravno 4 sekunde
                        leftServo.setPosition(100);
                        rightServo.setPosition(0);
                        Thread.sleep(4600);
                // zakrece se za priblizno 90 stupnjeva
                        leftServo.setPosition(100);
                        rightServo.setPosition(100);
                        Thread.sleep(625);

                        leftServo.off();
                        rightServo.off();
        }

        catch (Throwable t) {
        t.printStackTrace();
}
}
}
```